Neural networks in complex field

Li, Mingbin

2007

Li, M. (2007). Neural networks in complex field. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/46898>

<https://doi.org/10.32657/10356/46898>

Nanyang Technological University

Downloaded on 26 Apr 2023 22:15:48 SGT

Neural Networks in Complex Field

Li Mingbin

School of Electrical & Electronic Engineering

A thesis submitted to the Nanyang Technological University
in fulfilment of the requirement for the degree of
Doctor of Philosophy

2007

0A
76.87
L783
2007

Acknowledgments

First of all, I would like to express my sincere gratitude to my supervisor Assistant Professor Huang Guangbin for his brilliant guidance, encouragement and kindness during my three-year Ph.D research study. His abundant academic knowledge and instructive advice let me benefit a lot not only in research field but also in my life.

I would also like to thank Professor N. Sundadarajan and Associate Professor P. Saratchdran for their constructive suggestions and comments in my signal processing study and paper revision.

I am grateful to all staff and research students in ICIS lab, especially to Chen Lei and Zhu QinYu for their collaboration and help.

Special acknowledgments will be given to my parents, parents in law, my wife and my 9-month old daughter. Without their unselfish support in living and mind, I would not be able to achieve this success today.

Last but not least, I would like to show my thankfulness to School of Electrical and Electronic Engineering and Nanyang Technological University for giving me the opportunity to pursue Ph.D degree.

Summary

In digital communication systems, intersymbol interference (ISI) caused by multipath in time dispersive channels distorts the transmitted signal which is one of the major obstacles to high speed data transmission. Equalization is a technique used to combat intersymbol interference at receivers.

In the past decade, neural networks based equalizers have been thoroughly investigated. Many types of neural networks including multilayer feedforward neural networks, radial basis function (RBF) neural networks, minimal resource allocation networks (MRAN) and recurrent neural networks have been proposed successfully to solve channel equalization problems.

This thesis first extend growing and pruning RBF (GAP-RBF) network to the complex form, known as complex-valued GAP-RBF (CGAP-RBF) network, which can process the complex-valued input and output data. The communication channel equalization problem with QAM signal scheme is used to evaluate the performance of CGAP-RBF network. CGAP-RBF equalizer can achieve lower symbol error rate (SER) and faster learning speed than CRBF and CMRAN equalizers with more compact network topology.

In neural networks research, fast learning speed is another important fac-

tor for online application. Most of the existing learning algorithms are based on gradient-descent methods which adjust the network weights along the steepest gradient direction to obtain the optimal output at each step. These learning algorithms are usually very complicated and the training speed is very slow.

Extreme learning machine (ELM) proposed by Huang et al. is a novel learning algorithm compared with traditional gradient methods. The hidden nodes are randomly chosen and the output layer parameters are determined directly using a set of paired training data. Inspired by ELM's outstanding properties, we proposed the fully complex extreme learning machine (C-ELM), an extension of ELM in complex domain, which uses the fully complex activation functions in the hidden layer. The performance evaluation is presented in channel equalization to demonstrate that C-ELM is superior to other equalizers in terms of SER, learning speed and stability.

The approximation capability is an important property of neural networks. The performance of C-ELM learning algorithm is evaluated in two function approximation problems. Different activation functions have been used in C-ELM to compare their performances with other methods'. The simulation results show that C-ELM is an efficient learning algorithm. After that, the complex incremental extreme learning machine (C-IELM) is proposed. Furthermore, the approximation capability of C-IELM is theoretically proved.

Contents

Acknowledgments	i
Summary	ii
List of Figures	xiv
List of Tables	xv
1 Introduction	1
1.1 Motivation	1
1.2 Objectives	6
1.2.1 Complex-valued GAP-RBF neural networks	7
1.2.2 Complex incremental extreme learning machine	7
1.2.3 Fully complex extreme learning machine	8

1.3	Major Contribution of the Thesis	9
1.4	Organization of the Thesis	11
2	Literature Review	13
2.1	Complex-Valued Neural Networks	14
2.1.1	Complex-valued multilayer perceptron	14
2.1.2	Complex radial basis function (CRBF) neural networks	17
2.1.3	Complex minimal resource allocation network	20
2.2	Complex Domain Based Learning Algorithms	22
2.2.1	Complex backpropagation	22
2.2.2	Complex gradient descent learning algorithm for RBF networks	23
2.2.3	Complex extended Kalman filter for RBF networks . . .	25
2.3	Channel Equalization Problem	26
2.3.1	Digital communication system	26
2.3.2	Linear equalizer	28
2.3.3	Decision feedback equalizer	29

2.3.4	Bayesian equalizer	30
2.3.5	Neural networks based equalizers	31
2.4	Conclusions	33
3	Complex-Valued GAP-RBF Networks in Channel Equaliza-	
	tion	34
3.1	Introduction	35
3.2	CGAP-RBF Learning Algorithm	39
3.2.1	Significance of a hidden neuron	40
3.2.2	Uniform sampling distribution	42
3.2.3	Growing criterion	43
3.2.4	Pruning criterion	44
3.2.5	Parameter adjustment	45
3.3	Performance Evaluation of CGAP-RBF Equalizer	48
3.3.1	Third order complex linear channel model	49
3.3.2	Second order complex nonlinear channel model	54
3.3.3	Third order complex nonlinear channel model	57

3.3.4 Digital satellite channel model 62

3.3.5 Discussion 65

3.4 Conclusions 71

4 Complex Incremental Extreme Learning Machine 72

4.1 Introduction 73

4.2 Preliminaries 76

4.2.1 Review of I-ELM in real domain 76

4.2.2 Symbols and Theorems in complex domain 78

4.2.3 Necessary Lemmas 81

4.3 Approximation 82

4.4 Experimental Verification 85

4.4.1 Incremental Algorithm 86

4.4.2 Simulation 88

4.4.3 Discussion 91

5 Fully Complex Extreme Learning Machine 93

5.1 Introduction 94

5.2	Complex Extreme Learning Machine (C-ELM) Algorithm . . .	96
5.3	Performance Evaluation	100
5.3.1	Second order complex nonlinear channel model	100
5.3.2	Third order complex nonlinear channel model	104
5.4	Discussion	108
5.5	Conclusions	110
6	Function Approximation Using Complex Extreme Learning Machine	113
6.1	Introduction	114
6.2	C-ELM Algorithm with Different Activation Functions	116
6.2.1	Brief of complex extreme learning machine	116
6.2.2	Fully complex activation function	117
6.2.3	Radial basis activation function	118
6.3	Performance Evaluation	119
6.3.1	Non-analytic function approximation	120
6.3.2	Analytic function approximation	123

6.3.3 Multi-input and single output function 127

6.3.4 Discussion 128

6.4 Conclusions 130

7 Conclusion and Recommendations 131

7.1 Conclusions 131

7.2 Recommendations for Future Research 133

Author’s Publications 135

Bibliography 137

List of Figures

2.1	Multilayer feedforward neural networks	14
2.2	Split sigmoid function	15
2.3	Joint nonlinear function: $f(z) = z/(1 + z)$	16
2.4	$\operatorname{arcsinh}(z)$	17
2.5	Radial basis function network	19
2.6	Discrete model of communication system	27
2.7	Schematic of linear equalizer	28
2.8	Schematic of decision feedback equalizer	29
3.1	Channel output distribution (Example 1)	50
3.2	Growing of neurons (Example 1)	51
3.3	Training error (Example 1)	51

3.4 Error probability (Example 1) 52

3.5 Bayesian boundary (with 64 states) (Example 1) 53

3.6 CRBF boundary (with 30 states) (Example 1) 53

3.7 CMRAN boundary (with 45 states) (Example 1) 54

3.8 CGAP-RBF boundary (with 18 states) (Example 1) 54

3.9 Eye diagram of CRBF equalizer output (Example 2) 55

3.10 Eye diagram of CMRAN equalizer output (Example 2) 56

3.11 Eye diagram of CGAP-RBF equalizer output (Example 2) 57

3.12 Growing of neurons (Example 2) 57

3.13 Training error (CGAP-RBF equalizer) (Example 2) 58

3.14 Error probability for CRBF, CMRAN and CGAP-RBF (Ex-
ample 2) 58

3.15 Eye diagram of CRBF equalizer output (Example 3) 59

3.16 Eye diagram of CMRAN equalizer output (Example 3) 60

3.17 Eye diagram of CGAP-RBF equalizer output (Example 3) 60

3.18 Growing of neurons (Example 3) 61

3.19 Training error (Example 3) 61

3.20 Error probability for CRBF, CMRAN and CGAP-RBF (Ex-
ample 3) 62

3.21 Block diagram of the digital radio link with HPA (Example 4) 63

3.22 Growing of neurons (Example 4) 63

3.23 Training error for 16QAM signal (Example 4) 64

3.24 Error probability for CRBF, CMRAN, CGAP-RBF and ASNN
(Example 4) 65

4.1 Learning convergence for arcsin, arcsinh and arccos functions
(Example 5) 89

4.2 Training time for different functions (Example 5) 89

4.3 Learning convergence for arcsin, arcsinh and arccos functions
(Example 6) 90

4.4 Training time for different functions (Example 6) 91

5.1 Eye diagram of CBP equalizer output (Example 2) 101

5.2 Eye diagram of C-ELM equalizer output (Example 2) 102

5.3 Error probability for CRBF, CMRAN, CGAP-RBF, CBP and
C-ELM (Example 2) 102

5.4 Channel output distribution (Example 3) 104

5.5 Eye diagram of CRBF equalizer output (Example 3) 105

5.6 Eye diagram of CMRAN equalizer output (Example 3) 105

5.7 Eye diagram of CBP equalizer output (Example 3) 106

5.8 Eye diagram of C-ELM equalizer output (Example 3) 106

5.9 Error probability for C-ELM, CMRAN, CRBF and CBP (Ex-
ample 3) 107

5.10 Error probability for different hidden neuron (Example 3) . . . 111

6.1 Function output using C-ELM with joint activation function
(Example 5) 120

6.2 Function output using C-ELM with split activation function
(Example 5) 121

6.3 Function output using C-ELM with Gaussian activation func-
tion (Example 5) 121

6.4 Function output using CMRAN (Example 5) 122

6.5 Function output using CGAP-RBF (Example 5) 123

6.6 Function output using C-ELM algorithm (Example 6) 124

6.7 Function output using CBP algorithm (Example 6) 125

6.8 Function output using CMRAN (Example 6) 125

6.9 Function output using CGAP-RBF (Example 6) 126

List of Tables

3.1	Equalizer complexity for Example 3	68
3.2	Performance comparison of CGAP-RBF equalizer at different κ (Example 3).	70
5.1	Time comparisons of the five equalizers (a) C-ELM, (b) CBP, (c) CGAP-RBF, (d) CMRAN, (e) CRBF (Example 2)	103
5.2	Time comparisons of the four equalizers (a) C-ELM, (b) CBP, (c)CGAP-RBF, (d) CMRAN, (e) CRBF (Example 3)	108
5.3	Performance comparison of C-ELM equalizer with different hidden neuron (Example 3).	110
6.1	Performance comparison for Example 5	123
6.2	Performance comparison for Example 6	127
6.3	Performance comparison for Example 7	128

Chapter 1

Introduction

1.1 Motivation

Artificial Neural Network (ANN) is an information processing paradigm that is inspired by biological nervous systems [1, 2, 3]. Since McCulloch and Pitts [4] proposed the first neural model in 1943, many types of neural network with different learning algorithms have been developed, such as multilayer perceptron (MLP), recurrent neural network (RNN), self-organizing map (SOM) and radial basis function (RBF) neural network. The neural network is composed of a large number of highly interconnected processing elements (neurons) working in parallel to solve a specific problem. Because of its strong adaptive and approximation abilities, neural network has been applied to a wide variety of problems, including pattern recognition, function approxima-

tion, image compression, clustering, prediction, signal processing and control [5, 6, 7, 8, 9, 10, 11].

Among the various types of network models, multilayer perceptron (MLP) network is the most popular one [12, 13, 14, 15, 16, 17, 18]. Multilayer perceptron (MLP) network is known as a supervised network because it requires a desired output to train network weights and biases. The historical data is used to construct the network model which can map the input to the output correctly. Back-propagation algorithm is the common learning algorithm for MLP to adjust the weights and biases using error criteria iteratively. But multilayer perceptron model is highly nonlinear and suffers from slow convergence and unpredictable solution during learning process.

The radial basis function networks (RBF) have been attracted great interest from researchers due to their simple topological structure [19, 20, 21, 22, 23]. Being a single-hidden-layer network, the RBF network can implement a nonlinear mapping on an input by linearly combining outputs of the hidden nodes. The structures of the above mentioned neural network are fixed a priori based on designer's experience or trial and error. The networks may usually have larger number of hidden neurons, which will cause long computing time and reduce the efficiency of the neural network implementation.

Platt [24] proposes a growing criterion to allocate hidden neurons to net-

work based on the novelty in the observation that arrive sequentially, named resource allocation network (RAN). Kadiramanathan and Niranjan [25] use extended Kalman filter (EKF) rather than LMS algorithm to update RAN network parameters which improves the rate of convergence and reduces the network complexity. A new sequential learning algorithm, known as minimal resource allocation networks (MRAN), was developed by Yingwei et al [20, 26, 27]. MRAN adopts the basic idea of RAN-EKF and augments it with a pruning strategy to remove the hidden neurons that have insignificant contribution to the total output of the network. The drawback of MRAN is the difficulty to properly choose the window size of the growing and pruning algorithm, which can only be done by trial and error based on exhaustive simulation studies. Huang et al. [23] presented a sequential growing and pruning algorithm for RBF networks (GAP-RBF). The concept of “significance” for hidden neurons is used to evaluate the contribution of the hidden neurons to the overall performance of the RBF network. The proposed algorithm overcomes the drawbacks of MRAN and it can be universally used for a number of applications.

In fact, neural networks based equalizers have gained great interest as networks are well suit for nonlinear classification problem [28]. The channel transmitted symbols are distorted in both amplitude and phase known as intersymbol interference (ISI) and the additive noise in high speed digital communication systems. Equalizers are used very often at receivers to recover

the original symbols from the received signals, which can be simply divided into linear and non-linear equalization. Multi-Layer Perceptions (MLP), Radial Basis Function (RBF) networks and Recurrent Neural Networks (RNN) have been successfully used for equalization problems [29, 30, 31, 32, 33] and the performance of networks equalizers is superior to the linear equalizer in terms of bit error rate (BER) and computational complexity. Most of these networks based equalizers are applied for Pulse Amplitude Modulation (PAM) signal in real-valued field. Quadrature Amplitude Modulation (QAM) signals are complex-valued and the existing training algorithms of neural networks cannot work indeed. How to construct complex-valued networks and develop learning strategy to deal with complex-valued signals becomes very hot research topic for researchers in the past decade.

The complex-valued multilayer perceptron model has been proposed and the backpropagation algorithm is extended to a complex form. The “split” complex activation function is used to deal with the real and imaginary components which can be simply looked as two real-valued activation function [34, 35, 36, 37, 38]. Chen [39] proposes a complex-valued RBF networks for channel equalization and shows that its RBF network equalizer is structurally equivalent to the optimal Bayesian equalizer. Cha [40] gives an extension of RBF network for operation on complex signal in digital communication system. The adaptive stochastic gradient (SG) learning algorithm is used for adjusting the parameters of the RBF networks. The complex minimal re-

source allocation networks (CMRAN) is developed by Jianping et al. [41] for communication channel equalization with 4-QAM signal. The performance of CMRAN is compared with functional link artificial neural networks (FLANN) and complex-valued RBF (CRBF) networks and is superior in terms of symbol error rate (SER) and network complexity.

The sequential growing and pruning algorithm for RBF networks (GAP-RBF) proposed by Huang et al. [23] has been applied for function approximation. The performance of CAP-RBF network for channel equalization problem has not been investigated. As an efficient learning algorithm, its complex form should also be investigated and presented to solve the channel equalization problem with QAM signal.

Although the RBF networks have successfully used to deal with complex-valued data, the Euclidean distance between the input vector and center vector is still a real number. In real case, the activation functions are usually chosen the function that is analytic and bounded. However, according to complex function analysis, the main difficulty to find a suitable complex activation function is the conflict between the boundedness and the differentiability of complex function in the entire complex plane. In fact, a bounded entire function must be a constant in the complex plane, where the entire function means that the function is analytic every point in the complex plane. If the function is not analytic, there must be a singularity point at least, i.e., removable, isolated and essential singularities. The traditional way is use

“split” activation function which uses two real activation function to process the real and imaginary part of the complex number. Because the “split” activation function does not meet the Cauchy-Riemann conditions, it will compromise the efficiency of nonlinear approximation capability [42]. Kim and Adali propose a class of fully complex activation functions and apply it to channel equalization problem [37, 42]. While the backpropagation algorithm with fully complex activation function is sensitive to the value of the learning rate and the radius of initial random weights, the new learning algorithm should be investigated which can apply the fully complex activation function to deal with problems in complex domain.

Most of existed complex-valued research results are based on algorithm development and various applications. The theoretical proof about network approximation capability is few. How to theoretically prove the approximation capability of proposed networks is a very big challenge.

1.2 Objectives

The objectives of this research project are to develop the complex-valued growing and pruning RBF (GAP-RBF) network and study its performance in the communication channel equalization application. Another objective is to investigate the fully complex neural network and relevant learning algorithm. Then, the proposed fully complex neural network is evaluated in

the channel equalization and function approximation problems. Finally, the approximation capability of complex incremental extreme learning machine is proved theoretically.

1.2.1 Complex-valued GAP-RBF neural networks

Sequential learning algorithms have been investigated by many researchers in recent years. An efficient sequential learning algorithm named growing and pruning algorithm for RBF (GAP-RBF) networks is proposed by Huang et al. [23]. As we know, because real neural networks and learning strategy cannot be applied directly to complex-valued applications, some research works are needed to extend the real algorithm to complex form. Inspired by this motivation, we further investigate the complex form of the GAP-RBF network. The performance comparison with other sequential learning algorithms is carried out to show its superior abilities in terms of symbol error rate (SER) and network complexity in channel equalization with QAM signal scheme.

1.2.2 Complex incremental extreme learning machine

Recently, approximation capability of neural networks on complex domain has attracted considerable attention, because the approximation capability is an important property of neural networks. Recently an incremental algo-

rithm was proposed by Huang[43](I-ELM), which randomly chooses hidden nodes and then analytically determines the output weights connecting the hidden layer and the output layer. In this thesis, the complex incremental extreme learning machine (C-IELM) is proposed. Further, the theoretical proof is presented that feedforward neural network constructed by this algorithm can own universal approximation capability with randomly chosen hidden nodes.

1.2.3 Fully complex extreme learning machine

For complex-valued neural networks, it is hard to find a nonlinear complex activation function because of the conflict between the boundedness and the differentiability in the complex domain. Most of complex activation functions are chosen as one bounded but not entire function. Kim et al. [37, 42] derive a class of elementary transcendental functions (ETFs) as activation functions (entire but not bounded) for complex backpropagation algorithm, but the initial weights and biases need be chosen carefully to guarantee the backpropagation training convergence. Therefore, the new efficient learning algorithm should be investigated. The extreme learning machine with fully complex activation function can cover the drawback of complex backpropagation algorithm and can complete the learning phase in an extremely fast speed and lower symbol error rate.

1.3 Major Contribution of the Thesis

The major contributions of this thesis can be summarized as follows:

- A feedforward neural network using complex incremental extreme learning machine algorithm is proposed in complex domain, which randomly chooses hidden nodes and then analytically determines the output weights connecting the hidden layer and the output layer. Furthermore, its approximation capability is investigated and proved theoretically.
- Complex-valued growing and pruning algorithm for radial basis function (CGAP-RBF) network is proposed to solve the communication channel equalization problem with Quadrature Amplitude Modulation (QAM) signals scheme. By linking the significance of a neuron to the learning accuracy, a growing and pruning strategy for a radial basis function neural network with complex inputs is derived. Further, only the parameters of the nearest hidden neuron are adjusted during the pruning and growing strategy. The performance of the CGAP-RBF equalizer is compared with several other equalizers such as CMRAN, CRBF and ASNN on several nonlinear complex channel equalization problems. The simulation results show that the CGAP-RBF equalizer is superior to other equalizers in terms of symbol error rate and network complexity.

- A new learning algorithm for the feedforward neural network named the extreme learning machine (ELM) has been proposed by Huang, et al [44, 45], which can give better performance than traditional tuning-based learning methods for feedforward neural networks in terms of generalization and learning speed. In this thesis, we first extend the ELM algorithm from the real domain to the complex domain, and then apply the fully complex extreme learning machine (C-ELM) to the non-linear channel equalization applications. The simulation results demonstrate that the C-ELM equalizer significantly outperforms other neural network equalizers such as the complex minimal resource allocation network (CMRAN), complex radial basis function (CRBF) network and complex backpropagation (CBP) equalizers. C-ELM achieves much lower symbol error rate (SER) and has faster learning speed.
- The function approximation capability of C-ELM is investigated. C-ELM with different activation function is discussed in detail and applied to non-analytic and analytic function approximation. The simulation results show that C-ELM is an efficient method to deal with such problems.

1.4 Organization of the Thesis

This thesis consists of two parts. Part I is concerned with investigation of the complex sequential learning algorithm named CGAP-RBF networks and its applications; Part II presents the fully complex extreme learning machine and its application in channel equalization. The universal approximation capability of C-IELM is also investigated.

Chapter 2 summarizes the main research results in complex-valued neural networks and introduces the background of channel equalization problems.

Chapter 3 focuses on the derivation of the complex-valued GAP-RBF (CGAP-RBF) networks. The communication channel equalization with QAM signal is used to investigate the CGAP-RBF equalizer's performance. CGAP-RBF equalizer can obtain better performance than other network equalizers in terms of symbol error rate (SER) and equalizer complexity.

Chapter 4 proposes the complex incremental extreme learning machine and investigates its universal approximation capability. The theoretical proof is also given in this chapter.

Chapter 5 considers the fully complex extreme learning machine. Different with RBF kernel, the input, output, hidden neuron output and weights of networks are all complex-valued. Compared with traditional learning algorithm, the C-ELM algorithm can achieve fast learning speed.

1.4. Organization of the Thesis

12

Chapter 6 evaluates the C-ELM's function approximation capability in non-analytic and analytic functions. Several activation functions are chosen to implement C-ELM for approximation. C-ELM outperforms other methods in terms of learning accuracy and speed.

Chapter 7 summarizes the thesis and proposes some suggestions for the future work.

Chapter 2

Literature Review

The power and usefulness of artificial neural networks have been demonstrated in many applications including speech synthesis, diagnostic problems, business and finance, robotic control, signal processing, computer vision and many other problems [46, 47, 48, 49, 50, 51, 52]. Neural networks have the ability to represent both linear and non-linear relationships from the data being modeled. For some application areas, neural networks demonstrate more promising human-like performance over the traditional techniques.

In this chapter, we will focus on the reviews of neural networks in complex-valued domain and introduce the fundamental knowledge in channel equalization, including channel model and equalization methods.

2.1 Complex-Valued Neural Networks

2.1.1 Complex-valued multilayer perceptron

Neural networks process information in the way similar to what the human brain does. A number of interconnected neurons are working in parallel to solve a complicated problem. Multilayer feedforward neural networks is the most popular network structure and has been applied in many application fields [53, 54, 55, 56, 57, 58, 59, 60, 61]. Many studies showed that multilayer feedforward neural networks can form convex and nonconvex decision regions because of their nontrivial mapping capabilities [28]. A graphical representation of a MLP is shown in Fig. 2.1.

Figure 2.1: Multilayer feedforward neural networks

2.1. Complex-Valued Neural Networks

15

As analyzed in literature [34, 35], it may be hard to find a nonlinear complex activation function in network design in the entire complex plane. The traditional way is to use “split” complex activation as shown in formula (2.1), which has two real-valued activation functions for the real and imaginary components [13, 34, 36, 62]. This kind of “split” activation function usually employs sigmoid function as the complex activation function in a feedforward neural network structure:

$$f(z) = f_R(x) + if_I(y), \quad x, y \in R \quad (2.1)$$

where $f_R(x) = 1/(1 + \exp(-x))$ and $z = \mathbf{w} \cdot \mathbf{x} + b = x + iy$, where $\mathbf{x}$ is the complex-valued input vector, $\mathbf{w}$ is the complex-valued weight vector and b is the complex bias. Fig. 2.2 is shown the magnitude and phase of “split” sigmoid function. It can be seen that the “split” sigmoid function is bounded (maximum magnitude is $\sqrt{2}$) but not analytic.

Figure 2.2: Split sigmoid function

2.1. Complex-Valued Neural Networks

16

Georgiou and Koutsougeras [35] proposed another complex activation function.

$$f(z) = \frac{z}{c + \|z\|/r} \quad (2.2)$$

where c and r are real positive constants. The magnitude and phase of this function are given in Fig. 2.3. The function is bounded and has one removable singular point at $z = 0$.

Figure 2.3: Joint nonlinear function: $f(z) = z/(1 + |z|)$

Nine fully complex nonlinear activation functions have been proposed by Kim and Adali [37, 42, 63], which include circular functions ($\tan(z) = \frac{e^{iz} - e^{-iz}}{i(e^{iz} + e^{-iz})}$, $\sin(z) = \frac{e^{iz} - e^{-iz}}{2i}$), inverse circular functions ($\arctan(z) = \int_0^z \frac{dt}{1+t^2}$, $\arcsin(z) = \int_0^z \frac{dt}{(1-t^2)^{1/2}}$, $\arccos(z) = \int_0^z \frac{dt}{(1-t^2)^{1/2}}$), hyperbolic functions ($\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$, $\sinh(z) = \frac{e^z - e^{-z}}{2}$) and inverse hyperbolic functions ($\operatorname{arctanh}(z) = \int_0^z \frac{dt}{1-t^2}$, $\operatorname{arcsinh}(z) = \int_0^z \frac{dt}{(1+t^2)^{1/2}}$), where $z \in C$.

For the sake of brevity, we just discuss the properties of inverse hyper-

bolic functions $\operatorname{arcsinh}(z)$. Fig. 2.4 is the magnitude and phase of function $\operatorname{arcsinh}(z)$. The $\operatorname{arcsinh}$ function not only has removable singular at original point, but also has a branch cut along the imaginary axes. It can be seen that $\operatorname{arcsinh}(z)$ function is magnitude symmetric along the real and imaginary axes and the magnitude growth rate decreases with the input magnitude increasing. It is noted that the radial symmetry and the nonlinear phase response of $\operatorname{arcsinh}(z)$ tends to provide efficient nonlinear approximation capability [42].

Figure 2.4: $\operatorname{arcsinh}(z)$

2.1.2 Complex radial basis function (CRBF) neural networks

In recent years, radial basis function networks have gained much attention in sequential learning due to its simple topology structure. Several learning

algorithms have been proposed for complex-valued field in literature [49, 50, 64, 65, 66]. Compared with batch learning, sequential learning algorithms have the following distinguishing features [67]:

- The training observations will be *sequentially* presented to the learning system.
- At any time, *only one* training observation is learned.
- The learning system has no *prior* knowledge as to how many total training observations will be presented.
- A training observation is discarded as soon as the learning procedure for that particular observation is completed.

With the development of RBF networks, the complex-valued RBF networks have been investigating in the past decade. From the network structure point of view, the complex radial basis function network proposed by Chen et al. [39] is similar to the optimal Bayesian equalizer. Cha and Kassam [40] propose another complex radial basis function network referred to CRBF and use stochastic gradient algorithm to adjust the network parameters. In these complex RBF networks, both the inputs and outputs are complex value. The network can be viewed as a nonlinear mapping from the complex multi-dimensional input space to the complex multi-dimensional output space [39, 40, 68].

2.1. Complex-Valued Neural Networks

19

Fig. 2.5 illustrates a RBF networks structure with single output node.

The output of the CRBF networks can be expressed as follows:

Figure 2.5: Radial basis function network

$$f(\mathbf{x}) = \sum_{k=1}^K \alpha_{kR} \phi_k(\mathbf{x}) + j \left(\sum_{k=1}^K \alpha_{kI} \phi_k(\mathbf{x}) \right) = \sum_{k=1}^K \alpha_k \phi_k(\mathbf{x}) \quad (2.3)$$

where $\mathbf{x}$ is the n -dimensional complex input vector and α_k is the weight connecting the k -th hidden neuron to the output neuron. ϕ_k is Gaussian function of k -th hidden neuron and is given by

$$\phi_k(\mathbf{x}) = \exp\left(-\frac{1}{\sigma_k^2} (\mathbf{x} - \mu_k)^H (\mathbf{x} - \mu_k)\right) \quad (2.4)$$

where σ_k is the width of the k -th hidden neuron which is a real variable and μ_k is the n -dimensional complex center vector for the k -th hidden neuron. H denotes the complex conjugate transposition and $(\mathbf{x} - \mu_k)^H (\mathbf{x} - \mu_k)$ is the Euclidean distance between the input vector and center vector.

2.1.3 Complex minimal resource allocation network

A significant contribution to sequential learning algorithm was made by Platt [24], known as resource allocation network (RAN). The network starts with no hidden units and grows by allocating new hidden neurons based on the “novelty” of the input data. Enhancement of RAN, known as RAN-EKF was proposed by Kadirkamanathan and Niranjan [25] in which extended Kalman filter (EKF) rather than LMS algorithm was used for updating the network parameters-centers, widths and weights of Gaussian neurons. However it has the drawback of an increase in computational complexity. The RAN and RANEKF can only add neurons to the network and cannot prune insignificant neurons from the network. Because of this, both RAN and RANEKF could produce networks in which some hidden units, although active initially, may subsequently end up contributing little to the network output. If such inactive hidden units can be detected and removed as learning progresses, then a more parsimonious network topology can be realized.

To overcome this problem, a new sequential learning algorithm, known as minimal resource allocation networks (MRAN), was developed by Yingwei et al.[20, 26, 27]. MRAN adopts the basic idea of RANEKF and augments it with a pruning strategy to remove the hidden neurons that have insignificant contribution to the total output of the network. An additional growth criterion based on the output error over a sliding window is introduced to

smooth the transitions in the growth and pruning. Jianping et al. [41, 69] proposed the complex-valued minimal RAN network (CMRAN) for channel equalization problem, which is an extension of real MRAN algorithm. The applications of CMRAN for different nonlinear QAM channel equalization problem are investigated.

The growing and pruning parameters of CMRAN have to be chosen by trial and error based on exhaustive simulation studies. Huang et al. [23] presented a sequential growing and pruning algorithm for RBF networks (GAP-RBF). The algorithm makes use of the concept of “significance” for hidden neurons which is defined as the contribution of the hidden neurons to the overall performance of the RBF network. By linking the significance of the nearest neuron to the learning accuracy, a growing and pruning criterion is derived. Using these growing and pruning criteria, a more compact network structure is realized. Based on Huang’s work, we proposed a complex-valued GAP-RBF algorithm which is an extension of real GAP-RBF algorithm. The detail introduction will be given in the following chapter.

2.2 Complex Domain Based Learning Algorithms

With the development of complex-valued neural networks, the corresponding learning algorithms in complex domain have also investigated recently. In this section, we will briefly give an introduction for the existed learning algorithm to MLP and RBF networks.

2.2.1 Complex backpropagation

The complex backpropagation learning algorithm for multilayer neural networks has been proposed by many researchers in the past decade [34, 36, 42, 70, 71]. The cost function for multilayer neural networks is given by

$$E(k) = \frac{1}{2}[e^*(k)e(k)] \quad (2.5)$$

where $e(k) = d(k) - y(k)$ is the error between the target output $d(k)$ and network output $y(k)$ and $e^*(k)$ is the complex conjugate of $e(k)$. The network output $y(k)$ can be calculated by

$$y(k) = f(net_{l,n}(k))$$

$$net_{l,n}(k) = \sum_{m=1}^{K_{l-1}} x_{l-1,m}(k)w_{l,n,m}(k) \quad (2.6)$$

where $f(\cdot)$ is the complex activation function, K_{l-1} is the neuron number in layer $(l-1)$ ($0 \leq l \leq L$), L is the layer number of the networks, $x_{l-1,m}$ denotes

2.2. Complex Domain Based Learning Algorithms

23

the input from neuron m in layer $(l - 1)$, $w_{l,n,m}$ is the weight connecting neuron m in layer $(l - 1)$ to neuron n in layer l , respectively. The weights are adjusted by complex backpropagation algorithm giving bellow

$$w_{l,n,m}(k+1) = w_{l,n,m}(k) + \Delta w_{l,n,m}(k) \quad (2.7)$$

$$\Delta w_{l,n,m}(k) = \begin{cases} \eta(k)x_{0,m}^* \delta_{l,n}(k), & \text{if neuron } m \text{ in the input layer} \\ \eta(k)y_{l,m}^* \delta_{l,n}(k), & \text{otherwise} \end{cases} \quad (2.8)$$

The gradient components $(\delta_{l,n}(k))$ are defined by

$$\delta_{l,n}(k) = (f'(net_{l,n}(k)))^* e(k), \quad l = L \quad (2.9)$$

$$\delta_{l,n}(k) = (f'(net_{l,n}(k)))^* \sum_n^{K_{l+1}} w_{l+1,n,m}^* \delta_{l+1,n}(k), \quad 0 \leq l < L \quad (2.10)$$

It can be seen that the complex backpropagation algorithm is an extension of the real backpropagation. In order to deal with complex-valued data, the complex conjugate operation is used for the gradient components calculation. The real backpropagation can be looked as a special case of complex backpropagation algorithm.

2.2.2 Complex gradient descent learning algorithm for RBF networks

The radial basis function (RBF) networks with one single hidden layer has the ability of approximating nonlinear mapping [39, 40, 72, 73, 74]. The stochastic gradient (SG) descent for error criterion is adapted to modulate

2.2. Complex Domain Based Learning Algorithms

24

the network free parameters. Though SG algorithm cannot guarantee convergence to globally optimum network parameters, it will be convergent to reasonable solutions in practice [40]. For a RBF network with K hidden neurons, the output of networks is

$$f(\mathbf{x}_n) = \sum_{k=1}^K w_{k,n} \phi_k(\mathbf{x}_n) \quad (2.11)$$

where

$$\phi_k(\mathbf{x}_n) = \exp\left(-\frac{(\mathbf{x}_n - \mu_{k,n})^H (\mathbf{x}_n - \mu_{k,n})}{\sigma_{k,n}^2}\right) \quad (2.12)$$

The SG algorithm for parameter adjustment of complex RBF networks is given by

$$w_{k,n+1} = w_{k,n} + \eta_w e_n \phi_k(\mathbf{x}_n) \quad (2.13)$$

$$\begin{aligned} \sigma_{k,n+1} = & \sigma_{k,n} + \eta_\sigma \phi_k(\mathbf{x}_n) \\ & \cdot [\text{Re}(w_{k,n})\text{Re}(e_n) + \text{Im}(w_{k,n})\text{Im}(e_n)] \cdot \frac{(\mathbf{x}_n - \mu_{k,n})^H (\mathbf{x}_n - \mu_{k,n})}{\sigma_{k,n}^3} \end{aligned} \quad (2.14)$$

$$\begin{aligned} \mu_{k,n+1} = & \mu_{k,n} + \eta_\mu \phi_k(\mathbf{x}_n) \\ & \cdot \left[\frac{\text{Re}(w_{k,n})\text{Re}(e_n)\text{Re}(\mathbf{x}_n - \mu_{k,n}) + j\text{Im}(w_{k,n})\text{Im}(e_n)\text{Im}(\mathbf{x}_n - \mu_{k,n})}{\sigma_{k,n}^2} \right] \end{aligned} \quad (2.15)$$

where η_w, η_σ and η_μ are the adaptation coefficients for weight, width and center of complex RBF networks, respectively. $\text{Re}(\cdot)$ and $\text{Im}(\cdot)$ denote the real and imaginary component of complex-valued variable and $j = \sqrt{-1}$.

2.2.3 Complex extended Kalman filter for RBF networks

The drawback of gradient descent learning algorithm is time consuming, which inspires the alternative of extended Kalman filter (EKF) algorithm to adjust the RBF network parameters (weights, centers and widths) [25, 75, 76, 77]. The EKF method can improve the learning accuracy and obtain a more compact network. Many researches have been done to extend the EKF algorithm to complex domain in order to apply it to complex network parameters directly [41, 57, 78, 79]. The network parameter vector of complex RBF network with K hidden neurons is defined by

$$W = [\text{Re}(\alpha_0), \text{Im}(\alpha_0), \text{Re}(\alpha_1), \text{Im}(\alpha_1), \text{Re}(\mu_1^T), \text{Im}(\mu_1^T), \sigma_1, \dots, \text{Re}(\alpha_K), \text{Im}(\alpha_K), \text{Re}(\mu_K^T), \text{Im}(\mu_K^T), \sigma_K]^T \quad (2.16)$$

During the training process, the adjustment of network parameter vector is given below:

$$W_n = W_{n-1} + k_n [\text{Re}(e_n), \text{Im}(e_n)]^T \quad (2.17)$$

where k_n is the Kalman gain vector which is calculated by the following equations

$$k_n = P_{n-1} a_n [R_n + a_n^T P_{n-1} a_n]^{-1} \quad (2.18)$$

2.3. Channel Equalization Problem

26

$$\begin{aligned}
 a_n = & [I_{2 \times 2}, \phi_1(\mathbf{x}_n)I_{2 \times 2}, \phi_1(\mathbf{x}_n)(2\beta_1/\sigma_1^2) \cdot [\text{Re}(\mathbf{x}_n - \mu_1)^T, \text{Im}(\mathbf{x}_n - \mu_1)^T], \\
 & \phi_1(\mathbf{x}_n)(2\beta_1/\sigma_1^3)(\mathbf{x}_n - \mu_1)^H(\mathbf{x}_n - \mu_1), \dots, \phi_K(\mathbf{x}_n)I_{2 \times 2}, \\
 & \phi_K(\mathbf{x}_n)(2\beta_K/\sigma_K^2) \cdot [\text{Re}(\mathbf{x}_n - \mu_K)^T, \text{Im}(\mathbf{x}_n - \mu_K)^T], \\
 & \phi_K(\mathbf{x}_n)(2\beta_K/\sigma_K^3)(\mathbf{x}_n - \mu_K)^H(\mathbf{x}_n - \mu_K)]^T
 \end{aligned} \tag{2.19}$$

$$P_n = [I - k_n a_n^T] P_{n-1} + QI \tag{2.20}$$

where a_n is the gradient vector, P_n is the error covariance matrix and R_n is the covariance matrix of the measurement noise. $\beta_1 = [\text{Re}(\alpha_1), \text{Im}(\alpha_1)]^T, \dots, \beta_K = [\text{Re}(\alpha_K), \text{Im}(\alpha_K)]^T$, Q is a scalar that determines the allowed random step in the direction of gradient vector and I is the identity matrix.

2.3 Channel Equalization Problem

2.3.1 Digital communication system

Consider a standard baseband-equivalent model of a communication system (Fig. 2.6) where we assume that an i.i.d. M -ary signal sequence $\{s_n\}$ is transmitted through the channel and corrupted by additive zero-mean i.i.d. complex Gaussian noise $\{e_n\}$. The channel output y_n is given by

$$y_n = \sum_{i=1}^{n_h-1} h_i s_{n-i} + e_n \tag{2.21}$$

2.3. Channel Equalization Problem

27

where n_h is the channel order. A more general channel model gives

$$y_n = g(s_n, s_{n-1}, \dots, s_{n-n_h+1}) + e_n \quad (2.22)$$

Figure 2.6: Discrete model of communication system

$g(\cdot)$ is some complex nonlinear function. The task of the symbol decision equalizer is to estimate the transmitted decision delay $s_{n-\tau}$ based on noise channel observation vector $\mathbf{x}_n = [y_n, \dots, y_{n-m+1}]^T$ where m is the equalizer dimension and τ is the equalizer decision delay. The 4-QAM signal scheme is considered and it is given by $s_n = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$ where $\alpha_1 = a + ja$, $\alpha_2 = a - ja$, $\alpha_3 = -a + ja$, $\alpha_4 = -a - ja$. The noisy observation vector $\mathbf{x}_n$ is a random process with a conditional Gaussian function centered at each of the desired channel states. A minimum probability of error equalizer estimating the input symbol $s_{n-\tau}$ is given below

$$s_{n-\tau} = \alpha_j \quad (2.23)$$

$$j = \operatorname{argmax}\{h_p p(\mathbf{x}_n | s_{n-\tau} = \alpha_k), 1 \leq k \leq 4\}$$

where h_p is the prior occurrence probability of the input symbol α_k and p is the conditional probability density of the equalizer input state vector $\mathbf{x}_n = [y_n, \dots, y_{n-m+1}]^T$ conditioned on $s_{n-\tau} = \alpha_k$.

2.3.2 Linear equalizer

A linear equalizer can be implemented as an finite impulse response (FIR) filter [80, 81, 82, 83, 84]. The current and past received signals are linearly multiplied by the filter coefficients and summed to get the output, as shown in Fig. 2.7. If the length of the linear equalizer is m and decision delay is τ , the output of equalizer is

Figure 2.7: Schematic of linear equalizer

$$s_{out} = \sum_{k=0}^m w_k y_{n-k} \quad (2.24)$$

and

$$\hat{s}_{n-\tau} = \text{sgn}(\text{Re}(s_{out})) + j * \text{sgn}(\text{Im}(s_{out})) \quad (2.25)$$

where w_k is the equalizer coefficient and it is a complex-valued variable. The mean square error criterion is commonly used to adjust the coefficient w_k , which is to minimize the mean square error between the transmitted symbol

and the equalizer estimating output. However, the linear equalizer cannot obtain a very good error rate performance when the channel has deep spectral nulls, which caused the development of the nonlinear equalizers [46, 85, 86].

2.3.3 Decision feedback equalizer

Nonlinear equalizers are chosen in applications when the channel distortion is so severe that linear equalizer cannot deal with it. The decision feedback equalizer is a nonlinear equalizer and consists of a feedforward filter and a feedback filter [87, 88, 89, 90, 91, 92]. The structure of decision feedback equalizer is shown in Fig. 2.8. The equalizer output can be given by:

Figure 2.8: Schematic of decision feedback equalizer

$$s_{out} = \sum_{k=0}^m w_k y_{n-k} + \sum_{d=1}^N b_d \hat{s}_{n-\tau-d} \quad (2.26)$$

where b_d is the feedback filter's coefficient. The feedback filter is driven by equalizer detector decisions and its coefficients can be adjusted to reduce

2.3. Channel Equalization Problem

30

intersymbol interference (ISI) on the current symbol from the past detected symbols.

2.3.4 Bayesian equalizer

Because there are $n_s = 4^{n_h+m-1}$ combinations of the channel input sequence s_n , the noise-free channel output vector $\hat{\mathbf{x}}_n = [\hat{y}_n, \dots, \hat{y}_{n-m+1}]^T$, has n_s desired states and the set of these desired channel states, denoted as $R_{m,\tau}$, can be partitioned into four subsets according to the value of $s_{n-\tau}$:

$$R_{m,\tau} = \bigcup_{1 \leq k \leq 4} R_{m,\tau}^k \quad (2.27)$$

where

$$R_{m,\tau}^k = \{\hat{\mathbf{x}}_n | s_{n-\tau} = \alpha_k\}, 1 \leq k \leq 4 \quad (2.28)$$

The number of states in $R_{m,\tau}^k$ is $n_s^k = n_s/4$. Then a simple form of Bayesian decision function is given by

$$f_B(\mathbf{x}_n) = \sum_{k=1}^4 \sum_{q=1}^{n_s^k} h^k \exp \left(-\frac{(\mathbf{x}_n - \hat{\mathbf{x}}_q^k)^H (\mathbf{x}_n - \hat{\mathbf{x}}_q^k)}{2\sigma_e^2} \right) \quad (2.29)$$

where the inner sum is over $\hat{\mathbf{x}}_q^k \in R_{m,\tau}^k$, $h^k = \alpha_k$ is a complex signal, and symbol H means complex conjugate transposition. The Bayesian decision rule is

$$\hat{s}_{n-\tau} = \text{sgn}(\text{Re}(f_B(\mathbf{x}_n))) + j * \text{sgn}(\text{Im}(f_B(\mathbf{x}_n))) \quad (2.30)$$

The optimal decision boundary defined by

$$\{\mathbf{x} | f_B(\mathbf{x}) = 0 + j0\} \quad (2.31)$$

As observed from equation (2.29), the Bayesian equalizer is structurally equivalent to the complex radial basis function network. The RBF network weight α_k and the coefficients h^k of $f_B(\cdot)$ are pairs, the RBF centers μ_k correspond to the channel states $\hat{\mathbf{x}}_q$ and the width σ_k correspond to the noise variance σ_e [39, 93, 94].

2.3.5 Neural networks based equalizers

It is well known that channel equalization can be interpreted as a problem of nonlinear classification. Neural networks can learn the mapping relationship with a pair of received data which make it as an alternative to deal with classification problem [95, 96, 97]. Different types of neural networks have been found and successfully applied in the communication channel equalization problem [39, 66, 98, 99]. They have shown that because of the nonlinear characteristics of neural networks, the performance of neural network equalizers is superior to those of traditional linear and decision feedback equalizers (DFE).

Most popular neural networks [29, 46, 85, 100, 101] are good at processing real multi-dimensional signals in real domain and thus cannot be efficiently applied to communication channel equalizations with QAM signal directly. The new complex algorithms for neural networks should be investigated to deal with complex signal equalizers. The traditional multilayer

2.3. Channel Equalization Problem

32

networks with complex BP algorithm are developed by many research works [34, 35, 36, 37, 42]. Recently, radial basis function networks becomes an alternative for channel equalization because of its single hidden layer structure [39, 40]. All these network structures, the neuron number at each layer, are needed to decide a priori. Yingwei et al. [20, 26, 27] give a good attempt to realize a compact network structure by adding growing and pruning criterion to learn algorithm. The drawback of this method is that some parameters are hard chosen.

Recently, a growing and pruning RBF (GAP-RBF) algorithm has been proposed by Huang et al. [23]. Different from other sequential learning algorithms, GAP-RBF derives a very simply formula to calculate the significance of hidden neurons. By linking the required learning accuracy to the significance of neurons in the learning algorithm, a growing and pruning criterion is developed to realize a compact real valued RBF network. GAP-RBF have shown good performance on a number of benchmark problems in the function approximation area [23]. In this thesis, we will further investigate the complex form of GAP-RBF network and apply it to channel equalization problem.

2.4 Conclusions

In this chapter, some network knowledge in complex domain such as complex-valued neural networks and learning algorithm is briefly introduced first. The complex-valued networks are more complicated than the real one. The real domain networks can be considered as the special case of complex networks. For communication application, the channel model and popular equalization methods are also presented in this chapter.

Chapter 3

Complex-Valued GAP-RBF

Networks in Channel

Equalization

In this chapter, a complex-valued radial basis function network is proposed for solving the communication channel equalization problem with Quadrature Amplitude Modulation (QAM) signals. The network uses a sequential learning scheme named as complex growing and pruning (CGAP) algorithm, which is an extension of generalized growing and pruning (GGAP) algorithm developed by Huang et al. for real RBF networks. A very simple formula

⁰The main idea of this chapter can be found in "Complex-Valued Growing and Pruning RBF (CGAP-RBF) Neural Networks For Communication Channel Equalization," *IEE Proceedings on Vision, Image and Signal Processing*, Vol.153, no.4, pp.411-418, 2006

has been derived to calculate the significance of hidden neurons. By linking the neuron's significance to the learning accuracy, a growing and pruning strategy for radial basis function neural networks is proposed.

Furthermore, for both growing and pruning, one needs to check only the nearest neuron (based on the Euclidean distance to the latest input data) for its significance. When there is no growing or pruning, a complex extended Kalman filter is used to adjust the parameters of neural networks. The performance of the CGAP-RBF equalizer is compared with several other equalizers such as CMRAN, CRBF and ASNN in several nonlinear complex channel equalization problems. The results show that CGAP-RBF equalizer is superior to other equalizers in terms of symbol error rate and network complexity.

3.1 Introduction

It is well known that in high speed digital communication systems the channel distorts the transmitted symbols in both amplitude and phase thereby causing interference between adjacent symbols. This problem is compounded by the noise in the channel. Signal processing methods known as "equalization methods" are used at the receiver to recover the original symbols from the received signals.

Studies have shown that non-linear equalization methods perform better than linear methods [102], as they exploit some non-linearity in the equalization process. Maximum Likelihood (ML) strategies based on the Viterbi Algorithm (VA) [102] and its variants have been shown to provide the best performance among all equalization techniques. Generally ML algorithms require batch processing and are computationally intensive. Hence, equalizers that make decisions symbol-by-symbol have been gaining popularity. The optimal solution for these symbol-decision equalizers have been developed using the Bayesian decision theory [103, 104]. This Bayesian solution can be seen as a non-linear classification problem.

Multi-Layer Perceptions (MLP), Radial Basis Function (RBF) networks and Recurrent Neural Networks have been used for equalization problems [29, 85, 105] as neural networks are well suited for non-linear classification problems [28]. In QAM modulation schemas, the signals are complex-valued and hence the processing has to be done in a complex multi-dimensional space. The QAM modulation uses the data that have two components, amplitude and phase. Nonlinear characteristics of the channel cause spectral spreading, inter-symbol interference and constellation warping on the QAM signals. It is known that the QAM links are very sensitive to nonlinear distortion and the equalization of QAM signals is a difficult problem to solve [98]. For this type of important problems, in order to preserve the concise formulation and elegant structure of the complex signals, several complex

neural network equalizers have been proposed.

Chen et al. [39] proposed a complex radial basis function network, which is an extension of its real counterpart. The inputs and outputs of the network are both complex-valued while the radial basis function remains real. Furthermore, Cha and Kassam [40] proposed a complex RBF network (CRBF) equalizer using the stochastic gradient (SG) learning algorithm (with Gaussian basis function) to adjust all free parameters of the network simultaneously. Jianping et al. [41] developed a Complex MRAN equalizer for QAM channel equalization problems. They have also compared CMRAN with CRBF and have shown the better performance of CMRAN with reduced complexity. However, one of the problems in CMRAN is the difficulty in choosing the various thresholds for the growing and pruning part of the algorithm.

Uncini et al. [98] proposed a complex-valued neural network with an adaptable spline activation function using Catmull-Rom cubic splines to reduce the high complexity and long training time for equalizing a digital satellite radio link. In this link, the transmitter high-power amplifier (HPA) operates near the saturation and can cause serious degradation of the received signal. The nonlinearity of a typical HPA, which may be a traveling-wave tube (TWT) or a GaAs FET amplifier, affects both the amplitude and phase of the amplified signal. The resulting equalizer is a complex-valued neural network with adaptive spline activation function. Performance results of the

ASNN equalizer are shown to be comparable with finite order inverse volterra filters and to global compensation but with less complexity.

It should be noted that in the above equalizers, the equalization accuracy is not directly linked to the learning algorithm. Instead, they all have various thresholds which have to be selected using exhaustive trial-and-error studies.

Recently, a Generalized Growing and Pruning RBF (GGAP-RBF) algorithm has been proposed by Huang et al. [67] for real valued RBF networks. GGAP uses the concept of “significance” for the hidden neurons and directly links the required learning accuracy to the significance of neurons in the learning algorithm so as to realize a compact real valued RBF network. GGAP-RBF has shown better performance on a number of real world benchmark problems in the function approximation area [67].

In this chapter, we extend the notion of “significance” to complex valued inputs/outputs and use it for growing and pruning complex RBF equalizers. A new neuron will be added only if its significance is more than the chosen equalization accuracy. During the training, if a neuron’s significance is less than the equalization accuracy, that neuron will be pruned. Furthermore, for both growing and pruning, one needs to check only the nearest neuron (based on the Euclidean distance to the latest input data) for its significance. If the input observation does not require a new hidden neuron to be added, then the parameters of only the nearest neuron are adjusted, resulting in a

reduction in the overall computations and thereby increasing the equalization speed.

The performance of CGAP-RBF equalizer is compared with the CRBF [40], CMRAN [41] and the ASNN equalizer [98] in terms of symbol error rate and equalizer complexity. For the comparison with CRBF and CMRAN equalizers we have used the nonlinear channel models from Chen et al. [39] and Cha and Kassam [40]. For comparison with ASNN, we have used the channel model from Uncini et al. [98]. The results indicate the superior performance of CGAP-RBF equalizer in terms of symbol error rate and equalizer complexity.

3.2 CGAP-RBF Learning Algorithm

This section presents the newly proposed CGAP-RBF learning algorithm. First, following the real-space “Significance” concept introduced in Huang et al. [67], the concept of “Significance” for complex space is introduced and then a quantitative expression for it is derived. Using this significance, growing and pruning criteria for the hidden neurons are developed. When there is no growing or pruning, parameter adjustments using a complex EKF (extended Kalman filter) scheme are carried out.

3.2. CGAP-RBF Learning Algorithm

40

3.2.1 Significance of a hidden neuron

The output of a RBF network with K hidden neurons is given by

$$f(\mathbf{x}) = \sum_{k=1}^K \alpha_k \phi_k(\mathbf{x}), \quad \mathbf{x} \in \mathbf{C}^m \quad (3.1)$$

where $\phi_k(\mathbf{x})$ is the response of the k -th hidden neuron:

$$\phi_k(\mathbf{x}) = \exp \left(-\frac{(\mathbf{x} - \mu_k)^H (\mathbf{x} - \mu_k)}{\sigma_k^2} \right) \quad (3.2)$$

α_k is its connecting weight to the output neurons, and μ_k and σ_k are the center and width of the k -th hidden neuron, respectively, $k = 1, \dots, K$.

The network output at n -th observation $(\mathbf{x}_n, \mathbf{y}_n)$ is

$$f(\mathbf{x}_n) = \sum_{k=1}^K \alpha_k \phi_k(\mathbf{x}_n) \quad (3.3)$$

The training samples are randomly drawn from a range X with a joint density function $p(\mathbf{x})$, which is given by

$$\int \cdots \int_X p(\mathbf{x}) d\mathbf{x} = 1 \quad (3.4)$$

The probability density function $p(\mathbf{x})$ for a scalar complex variable X can be interpreted as the joint density function for two real variables $p_{xy}(X)$, in which the complex variable X is $X = x + jy$. If the neuron k is insignificant for the overall network output at observation $\mathbf{x}_t$, the error caused by removing the neuron k will be

$$E(k, t) = \|\alpha_k\|_2 \phi_k(\mathbf{x}_t) = \|\alpha_k\|_2 \exp \left(-\frac{(\mathbf{x}_t - \mu_k)^H (\mathbf{x}_t - \mu_k)}{\sigma_k^2} \right), \quad t = 1, \dots, n \quad (3.5)$$

3.2. CGAP-RBF Learning Algorithm

41

where $\|\cdot\|_2$ denotes the 2-norm of vectors. Then, the average error for all n sequentially learned observations caused by removing the neuron k will be

$$E_{ave}(k) = \left(\frac{\sum_{t=1}^n E^2(k, t)}{n} \right)^{1/2} = \|\alpha_k\|_2 \left(\frac{\sum_{t=1}^n \phi_k^2(\mathbf{x}_t)}{n} \right)^{1/2} \quad (3.6)$$

The size of the input sampling space is denoted by $S(X)$, $S(X) = \int \cdots \int_X 1 d\mathbf{x}$. Let the sampling area X be divided into N small areas $\Delta_d, d = 1, \dots, N$. The size of each small sub-space Δ_d is represented by $S(\Delta_d)$. Then the number of samples in each small area Δ_d can be estimated as $n \cdot p(\mathbf{x}_d) S(\Delta_d)$, where Y_d is any random chosen point in the area Δ_d . Therefore, we have

$$\begin{aligned} E_{ave}(k) &\approx \|\alpha_k\|_2 \left(\frac{\sum_{d=1}^N \phi_k^2(\mathbf{x}_d) n p(\mathbf{x}_d) S(\Delta_d)}{n} \right)^{1/2} \\ &\approx \|\alpha_k\|_2 \left(\sum_{d=1}^N \phi_k^2(\mathbf{x}_d) p(\mathbf{x}_d) S(\Delta_d) \right)^{1/2} \end{aligned} \quad (3.7)$$

When the number of input observations n is large enough such that the sub-area $S(\Delta_d)$ is sufficiently small, we have

$$\begin{aligned} \lim_{N \rightarrow \infty} E_{ave}(k) &= \lim_{N \rightarrow \infty} \|\alpha_k\|_2 \left(\sum_{d=1}^N \phi_k^2(\mathbf{x}_d) p(\mathbf{x}_d) S(\Delta_d) \right)^{1/2} \\ &\approx \|\alpha_k\|_2 \left(\int \cdots \int_X \exp \left(-\frac{2(\mathbf{x} - \mu_k)^H (\mathbf{x} - \mu_k)}{\sigma_k^2} \right) p(\mathbf{x}) d\mathbf{x} \right)^{1/2} \end{aligned} \quad (3.8)$$

$\lim_{N \rightarrow \infty} E_{ave}(k)$ is the average information content of neuron k , the contribution of hidden neuron k to the overall performance of the RBF network.

3.2. CGAP-RBF Learning Algorithm

42

Therefore, the “significance” of neuron k can be defined and quantified as

$$\begin{aligned} E_{sig}(k) &= \lim_{N \rightarrow \infty} E_{ave}(k) \\ &= \|\alpha_k\|_2 \left(\int \cdots \int_X \exp \left(-\frac{2(\mathbf{x} - \mu_k)^H (\mathbf{x} - \mu_k)}{\sigma_k^2} \right) p(\mathbf{x}) d\mathbf{x} \right)^{1/2} \end{aligned} \quad (3.9)$$

3.2.2 Uniform sampling distribution

When the sampling density function $p(\mathbf{x})$ is uniform, then it can be presented by $p(\mathbf{x}) = \frac{1}{S(X)}$, where $S(Y)$ is the size of the input sampling space. The equation (3.9) can be rewritten as

$$\begin{aligned} E_{sig}(k) &= \|\alpha_k\|_2 \left(\int \cdots \int_X \exp \left(-\frac{2(\mathbf{x} - \mu_k)^H (\mathbf{x} - \mu_k)}{\sigma_k^2} \right) \frac{1}{S(X)} d\mathbf{x} \right)^{1/2} \\ &= \frac{\|\alpha_k\|_2}{S(X)^{1/2}} \left(\int \cdots \int_X \exp \left(-\frac{2(\mathbf{x} - \mu_k)^H (\mathbf{x} - \mu_k)}{\sigma_k^2} \right) d\mathbf{x} \right)^{1/2} \end{aligned} \quad (3.10)$$

We express the input vector $\mathbf{x}_n = [y_1, y_2, \dots, y_m]$ by complex polar form $\mathbf{x}_n = [r_1 \exp(j\theta_1), r_2 \exp(j\theta_2), \dots, r_m \exp(j\theta_m)]$, where m is the input dimension (equalizer order), r_l and θ_l ($1 \leq l \leq m$) are the complex number modulus and phase angle of l -th dimensional input complex value y_l of input symbol $\mathbf{x}_n$, respectively. It is well known that for any complex number z , its phase angle θ can be expressed as $\theta = \text{Arg}(z) + 2k\pi, k = (0, \pm 1, \pm 2, \dots)$, where $\text{Arg}(z)$ is the principal value of the phase angle, defined by $-\pi < \text{Arg}(z) \leq \pi$. Here, the input vector phase angle θ_l is simply defined to have the principal

3.2. CGAP-RBF Learning Algorithm

43

value, i.e., $-\pi < \theta_l \leq \pi$ ($1 \leq l \leq m$), then

$$\begin{aligned} E_{sig}(k) &= \frac{\|\alpha_k\|_2}{S(X)^{1/2}} \prod_{i=1}^m \left(\int_{\theta_{\min}}^{\theta_{\max}} d\theta \int_0^\infty \exp\left(-\frac{2r^2}{\sigma_k^2}\right) dr \right)^{1/2} \\ &= \frac{\|\alpha_k\|_2}{S(X)^{1/2}} \left((\theta_{\max} - \theta_{\min}) \left(\frac{\sqrt{\pi}}{2\sqrt{2}} \sigma_k \right) \right)^{m/2} \end{aligned} \quad (3.11)$$

where $\theta_{\max}$ and $\theta_{\min}$ are the maximal and minimal principal values of the phase angle in the sample area of one dimension input. When the training data center around the origin $0 + j0$, $\theta_{\max}$ and $\theta_{\min}$ can be simply seen as $\theta_{\max} = \pi$ and $\theta_{\min} = -\pi$ for uniform distribution case. Thus, from formula (3.11) we have

$$E_{sig}(k) = \frac{\|\alpha_k\|_2}{S(X)^{1/2}} \left(\frac{\pi^{3/2}}{\sqrt{2}} \sigma_k \right)^{m/2} \quad (3.12)$$

3.2.3 Growing criterion

During training, if $E_{sig}(k) < e_{goal}$ where e_{goal} is the target accuracy, the average contribution made by the neuron k is less than the expected accuracy (here, the equalization accuracy). This means that neuron k is insignificant and it can be removed. Similarly, for a newly added neuron $K + 1$, if $E_{sig}(K + 1) < e_{goal}$, it means that the newly added neuron makes insignificant contribution to the overall performance of the network (although it may make sense to that single input observation). Thus, this neuron should not be added. An enhanced growing criterion is given by:

3.2. CGAP-RBF Learning Algorithm

44

For new observation $(\mathbf{x}_n, \mathbf{y}_n)$, if

$$\begin{cases} \|\mathbf{x}_n - \mu_{nr}\| > \epsilon_n \\ E_{sig}(K+1) > e_{goal} \end{cases} \quad (3.13)$$

where ϵ_n is a distance threshold for adding neurons, a new neuron $K+1$ should be added and the parameters associated with the new hidden neuron are selected as follows:

$$\begin{cases} \alpha_{K+1} = e_n \\ \mu_{K+1} = \mathbf{x}_n \\ \sigma_{K+1}^2 = \kappa(\mathbf{x}_n - \mu_{nr})^H(\mathbf{x}_n - \mu_{nr}) \end{cases} \quad (3.14)$$

where $e_n = \mathbf{s}_{n-\tau} - f(\mathbf{x}_n)$ and μ_{nr} is the center, which is nearest to $\mathbf{x}_n$. κ is an overlap factor that determines the overlap of the responses of the hidden neurons in the input space.

The first criterion ensures that a new neuron is only added if the input data is sufficiently far from the existing neurons. The second criterion ensures that the significance of the newly added neuron obtained is greater than the required approximation accuracy e_{goal} .

3.2.4 Pruning criterion

If the significance of neuron k is less than the accuracy e_{goal} , neuron k is insignificant and should be removed, otherwise, neuron k is significant and

3.2. CGAP-RBF Learning Algorithm

45

should be retained. The above condition implies that after learning each observation, the significance for all neurons should be computed and checked for possible pruning. This will be a computationally intensive task. However, only the nearest neuron can possibly be insignificant and need to be checked for pruning and there is no need to compute the significance for all neurons.

Similar to GGAP-RBF for real domain case [67], when no neuron is added or pruned from the network, in CGAP-RBF only the parameters of the neuron nearest to the latest incoming observation will be adjusted using the extended Kalman filter (EKF) algorithm.

3.2.5 Parameter adjustment

When no neuron is added or pruned from the network, the network parameter vector is adjusted using an extended Kalman filter (EKF) algorithm

$$W_n = W_{n-1} + k_n [\text{Re}(e_n), \text{Im}(e_n)]^T \quad (3.15)$$

where $W = [\text{Re}(\alpha_1), \text{Im}(\alpha_1), \text{Re}(\mu_1^T), \text{Im}(\mu_1^T), \sigma_1, \dots, \text{Re}(\alpha_K), \text{Im}(\alpha_K), \text{Re}(\mu_K^T), \text{Im}(\mu_K^T), \sigma_K]^T$ is the network parameters vector. $\text{Re}(\cdot)$ and $\text{Im}(\cdot)$ represent the real and imaginary parts, respectively. k_n is the Kalman gain vector given by

$$k_n = P_{n-1} a_n [R_n + a_n^T P_{n-1} a_n]^{-1} \quad (3.16)$$

3.2. CGAP-RBF Learning Algorithm

46

a_n is the gradient vector and has the following form:

$$\begin{aligned}
 a_n = & [0_{2 \times (nr-1) * (2 * input_dim + 2 * output_dim + 1)}, \phi_{nr}(\mathbf{x}_n) I_{2 \times 2}, \\
 & \phi_{nr}(\mathbf{x}_n) (2\beta_{nr} / \sigma_{nr}^2) \cdot [\text{Re}(\mathbf{x}_n - \mu_{nr})^T, \text{Im}(\mathbf{x}_n - \mu_{nr})^T], \\
 & \phi_{nr}(\mathbf{x}_n) (2\beta_{nr} / \sigma_{nr}^3) (\mathbf{x}_n - \mu_{nr})^H (\mathbf{x}_n - \mu_{nr}), \\
 & 0_{2 \times (K - nr) * (2 * input_dim + 2 * output_dim + 1)}]^T
 \end{aligned} \tag{3.17}$$

where $\beta_{nr} = [\text{Re}(\alpha_{nr}), \text{Im}(\alpha_{nr})]^T$.

Here, we only adjust the parameters of the nearest neuron. P_n is the error covariance matrix which is updated by

$$P_n = [I - k_n a_n^T] P_{n-1} + Q I \tag{3.18}$$

where Q is scalar that determines the allowed random step in the direction of gradient vector and I is the identity matrix. When a new neuron is allocated, the dimensionality of the P_n increase to

$$P_n = \begin{pmatrix} P_{n-1} & 0 \\ 0 & P_0 I \end{pmatrix} \tag{3.19}$$

The different steps involved in the CGAP-RBF algorithm are summarized as:

CGAP-RBF Algorithm

For each observation $(\mathbf{x}_n, \mathbf{y}_n)$, do

1. **compute** the overall network output:

$$f(\mathbf{x}_n) = \sum_{k=1}^K \alpha_k \exp \left(-\frac{(\mathbf{x}_n - \mu_k)^H (\mathbf{x}_n - \mu_k)}{\sigma_k^2} \right) \tag{3.20}$$

3.2. CGAP-RBF Learning Algorithm

47

where K is the number of hidden neurons.

2. **calculate** the parameters required in the growth criterion:

$$\begin{aligned}\epsilon_n &= \max\{\epsilon_{\max}\gamma^n, \epsilon_{\min}\}, \quad (0 < \gamma \leq 1) \\ e_n &= \mathbf{y}_n - f(\mathbf{x}_n)\end{aligned}\tag{3.21}$$

3. **apply** the criterion for adding a neuron:

If $\|\mathbf{x}_n - \mu_{nr}\| > \epsilon_n$ and $E_{sig}(K+1) > e_{goal}$, then

allocate a new hidden neuron $K+1$ with

$$\begin{aligned}\alpha_{K+1} &= e_n \\ \mu_{K+1} &= \mathbf{x}_n \\ \sigma_{K+1}^2 &= \kappa(\mathbf{x}_n - \mu_{nr})^H(\mathbf{x}_n - \mu_{nr})\end{aligned}\tag{3.22}$$

Else

adjust the network parameters α_{nr} , μ_{nr} , σ_{nr} for the nearest neuron only.

check the criterion for pruning the adjusted hidden neuron:

If $E_{sig}(nr) < e_{goal}$

remove the nr -th hidden neuron

reduce the dimensionality of EKF

Endif

Endif

γ ($0 < \gamma < 1$) in formula (3.21) is a decay constant and used to adjust the distance threshold ϵ_n so that it will decrease gradually from the maximum value $\epsilon_{\max}$ to $\epsilon_{\min}$, where $\epsilon_{\max}$ is the largest length of the input space and $\epsilon_{\min}$ is the desired distance between the hidden neurons. The key idea is to first find and cover the most troublesome position in the input space. Then, it is gradually reduced to cover the input space until it reach $\epsilon_{\min}$. This method can help to distribute the created hidden neuron in the all input space and $\epsilon_{\min}$ can avoid any of two neuron too close each other.

3.3 Performance Evaluation of CGAP-RBF Equalizer

The performance of the CGAP-RBF equalizer for QAM signals is evaluated for one linear and two nonlinear complex channel equalization benchmark problems.

First, a third order linear channel [39] is considered for 4-QAM signal equalization. The optimal decision boundary in this case is highly nonlinear. Here, the performance of CGAP-RBF is compared with CMRAN and the optimal Bayesian equalizer in terms of symbol error rate, network size and decision boundaries. Secondly, a nonminimum-phase channel with nonlinear distortion signal from Cha and Kassam [40] is used to evaluate the

performance of the CGAP-RBF equalizer with CMRAN and CRBF equalizers. Finally, a satellite radio channel with a high power amplifier (HPA) [98] is chosen for 16-QAM signal modulation scheme. In this case, the channel nonlinear distortion introduced by the transmitter HPA has a twofold effect: increasing of intersymbol interference and widening the spectrum of the transmitted signal to increase adjacent channel interference. The performance of CGAP-RBF equalizer for this model is studied and compared with the ASNN equalizer.

In all these studies, the CGAP-RBF equalizer uses formula (3.11) for significance derived for uniform input distribution although a general expression for significance for any distribution is given in formula (3.9). This is because the expression for significance for uniform input distribution is computationally simpler. It should be pointed out here that even though the channel inputs are uniformly distributed, the equalizer inputs will always be non-uniform. However, in all our studies, the CGAP equalizer based on uniform input distribution performed better than the other equalizers.

3.3.1 Third order complex linear channel model

Example 1: Here, a third order complex channel model as considered by Chen et al. [39] for 4-QAM signaling is used and is given by

$$A(z) = (0.7409 - j0.7406)(1 - (0.2 - j0.1)z^{-1})(1 - (0.6 - j0.3)z^{-1}) \quad (3.23)$$

3.3. Performance Evaluation of CGAP-RBF Equalizer

50

The noise variance is $\sigma_e^2 = 0.06324$ (SNR=15dB). The equalizer dimension was set to $m = 1$ and the equalizer decision delay was $\tau = 0$ same as in Chen et al. [39]. The CGAP-RBF network parameters are set as: $e_{goal} = 0.3$, $\epsilon_{\max} = 0.2$, $\gamma = 1$, $\kappa = 0.1$. For finding the size of S which is the range of the equalizer input, we need to look at the channel output (same as the equalizer input) and estimate it based on the area of the noisy channel output. The noisy channel output is shown in Fig. 3.1 and from this S can be estimated as $S = (4^{n_h} * \pi * 0.15^2)^m$ where 0.15 is the rough estimate of the radius of each sub-area.

Figure 3.1: Channel output distribution (Example 1)

The CGAP-RBF equalizer is trained using 2000 samples at 15dB SNR. The evolution of the hidden neurons during training and the training error for CGAP-RBF and CMRAN equalizers are given in Fig. 3.2 and Fig. 3.3, respectively. The CRBF equalizer is trained using 15000 samples with 30

Figure 3.2: Growing of neurons (Example 1)

Figure 3.3: Training error (Example 1)

hidden neurons. For testing, 100 000 test data of different SNRs were used and the symbol error rate was evaluated. Fig. 3.4 shows the comparison of the symbol error rate (SER) between Bayesian, CRBF, CMRAN and CGAP-RBF equalizers.

3.3. Performance Evaluation of CGAP-RBF Equalizer

52

Figure 3.4: Error probability (Example 1)

We can see that CGAP-RBF is always better than CRBF and CMRAN but is not as good as Bayesian method. As mentioned in section 2.3.4, Bayesian method uses all noise-free channel output states as its equalizer centers, which can achieve the best optimal equalization performance. However, The CGAP-RBF equalizer uses only 18 hidden neurons, which is much smaller than that of CRBF (30 neurons), CMRAN (45 neurons) and the Bayesian (64 neurons). It should be noted that the Bayesian equalizer requires the exact channel model and also is computationally more intensive.

For this problem, the performance of the different equalizers can be qualitatively seen from the decision boundary diagrams. Fig. 3.5 shows the decision boundary using the Bayesian equalizer and it partitions the observation space into four decision regions and this is a complicated boundary. Fig. 3.6-Fig. 3.8 show the same for CRBF, CMRAN and CGAP-RBF equalizer,

respectively.

Figure 3.5: Bayesian boundary (with 64 states) (Example 1)

Figure 3.6: CRBF boundary (with 30 states) (Example 1)

Figure 3.7: CMRAN boundary (with 45 states) (Example 1)

Figure 3.8: CGAP-RBF boundary (with 18 states) (Example 1)

3.3.2 Second order complex nonlinear channel model

Example 2: A second order complex nonlinear channel model [40] is used here to test the CGAP-RBF equalizer performance. The overall channel

output is given by

$$\begin{aligned} y_n &= o_n + 0.2o_n^2 + 0.1o_n^3 + v_n, \quad v_n \sim N(0, 0.01) \\ o_n &= (1 - j0.3434)s_n + (0.5 + j0.2912)s_{n-1} \end{aligned} \quad (3.24)$$

The CGAP-RBF equalizer dimension was set to $m = 2$ and the equalizer decision delay was $\tau = 0$. The CGAP-RBF network parameters are set as: $e_{goal} = 0.2, \epsilon_{max} = 0.2, \gamma = 1, \kappa = 0.2$. The range of input sample S is estimated using $S = (4^{n_h} * \pi * 0.2^2)^m$.

Figure 3.9: Eye diagram of CRBF equalizer output (Example 2)

CGAP-RBF and CMRAN equalizers are trained with 1000 data samples at 20dB SNR and CRBF equalizer is trained with 30 hidden neurons and 10000 samples. Fig. 3.9-Fig. 3.11 are the eye diagrams CRBF, CMRAN and CGAP-RBF equalizers. Though all equalizers can divide the space into four group, CGAP-RBF equalizer is more clear than CMRAN and CRBF

Figure 3.10: Eye diagram of CMRAN equalizer output (Example 2)

equalizers.

The growing of neurons for CGAP-RBF and CMRAN equalizers is shown in Fig. 3.12. At the end of training process, CGAP-RBF and CMRAN equalizers build up 30 and 32 hidden neurons, respectively. The curves for training error are given in Fig. 3.13.

1000 000 test samples are used for SER calculation. The SER curves for CRBF, CMRAN and CGAP-RBF equalizers are displayed in Fig. 3.14. With 30 hidden neuron and 1000 training samples, CGAP-RBF equalizer achieve better SER performance at various SNRs than the other two equalizers.

Figure 3.11: Eye diagram of CGAP-RBF equalizer output (Example 2)

Figure 3.12: Growing of neurons (Example 2)

3.3.3 Third order complex nonlinear channel model

Example 3: For this case, the complex nonminimum-phase channel model with nonlinear distortion as used in Cha and Kassam [40] is used to evaluate

3.3. Performance Evaluation of CGAP-RBF Equalizer

58

Figure 3.13: Training error (CGAP-RBF equalizer) (Example 2)

Figure 3.14: Error probability for CRBF, CMRAN and CGAP-RBF (Example 2)

the CGAP-RBF equalizer performance. The channel output is given by

$$y_n = o_n + 0.1o_n^2 + 0.05o_n^3 + v_n, \quad v_n \sim N(0, 0.01) \quad (3.25)$$

$$o_n = (0.34 - j0.27)s_n + (0.87 + j0.43)s_{n-1} + (0.34 - j0.21)s_{n-2}$$

3.3. Performance Evaluation of CGAP-RBF Equalizer

59

The CGAP-RBF equalizer dimension was set to $m = 3$ and the equalizer decision delay was $\tau = 1$, same as chosen in Cha and Kassam [40]. The CGAP-RBF network parameters are set as: $e_{goal} = 0.2$, $\epsilon_{max} = 0.2$, $\gamma = 1$, $\kappa = 0.2$. The range of input sample S is estimated using $S = (4^{n_h} * \pi * 0.2^2)^m$.

CGAP-RBF equalizer is trained with 1000 data symbols at 20dB SNR.

Figure 3.15: Eye diagram of CRBF equalizer output (Example 3)

The eye diagrams for CRBF, CMRAN and CGAP-RBF equalizers are shown in Fig. 3.15, Fig. 3.16 and Fig. 3.17. It is worthy noting that CRBF equalizer used 13,500 training samples while CMRAN used 1400 and CGAP-RBF only 1000.

It can be seen from the figures that although three equalizers can separate the data into four zones clearly, the CGAP-RBF equalizer needs less training samples than CMRAN and CRBF equalizers. CGAP-RBF obvi-

Figure 3.16: Eye diagram of CMRAN equalizer output (Example 3)

Figure 3.17: Eye diagram of CGAP-RBF equalizer output (Example 3)

ously converges faster than other algorithms. Further, for this problem, the CGAP-RBF equalizer uses only 18 hidden neurons, which is much smaller than that of CMRAN (25 neurons) and the CRBF (30 neurons).

Figure 3.18: Growing of neurons (Example 3)

Figure 3.19: Training error (Example 3)

Fig. 3.18 shows the growth of the neurons (CGAP-RBF and CMRAN equalizers) and Fig. 3.19 shows the training error.

Test sets with 100 000 samples at various SNRs were used for computing

3.3. Performance Evaluation of CGAP-RBF Equalizer

62

the SER and a comparison of SER for three equalizers is shown in Fig. 3.20. From the figure, it can be seen that CGAP-RBF produces the least SER in all equalizers.

Figure 3.20: Error probability for CRBF, CMRAN and CGAP-RBF (Example 3)

3.3.4 Digital satellite channel model

Example 4: In this problem, the digital radio link (satellite) channel model of Uncini et al. [98] for a 16-QAM signal is used for CGAP-RBF evaluation.

In this link, the transmitter HPA operates near the saturation and can cause serious degradation of the received signal. The nonlinearity of a typical HPA, which may be a traveling-wave tube (TWT) or a GaAs FET amplifier,

3.3. Performance Evaluation of CGAP-RBF Equalizer

63

Figure 3.21: Block diagram of the digital radio link with HPA (Example 4)

Figure 3.22: Growing of neurons (Example 4)

affects both the amplitude and phase of the amplified signal. The block diagram of the digital radio link with HPA is shown in Fig. 3.21 and the model for the HPA is given by [98]

$$v(t) = \frac{2u(t)}{1 + |u(t)|^2} \exp \left[j\Phi_0 \frac{2|u(t)|^2}{1 + |u(t)|^2} \right] \quad (3.26)$$

3.3. Performance Evaluation of CGAP-RBF Equalizer

64

where the $u(t)$ is the output of the pulse shaping filter and Φ_0 is the maximum shift of HPA. The shaping filter $g(t)$ is chosen same as in paper [98].

The CGAP-RBF equalizer input dimension was set to $m = 9$ and the equalizer decision delay was $\tau = 0$. The CGAP-RBF network parameters are set as: $e_{goal} = 0.2$, $\epsilon_{max} = 0.2$, $\gamma = 1$, $\kappa = 0.1$, $S = (16 * \pi * 0.1^2)^m$.

Figure 3.23: Training error for 16QAM signal (Example 4)

In this case, 5000 training samples are used to train the CGAP-RBF and CMRAN equalizers at 20dB SNR. 15000 training samples are used for CRBF equalizer. The training error and the growth progress of the neurons in the hidden layer during the training for CGAP-RBF and CMRAN equalizers are given in Fig. 3.22 and Fig. 3.23, respectively. 100 000 test data at various SNRs were used for the SER evaluation. SER comparison between CRBF, CMRAN, CGAP-RBF and ASNN is given in Fig. 3.24, where the results of

Figure 3.24: Error probability for CRBF, CMRAN, CGAP-RBF and ASNN (Example 4)

ASNN were referred from Uncini et al. [98] and 100 000 different training symbols (or training iterations) were used for ASNN. It can be seen from the figure that CGAP-RBF equalizer produces smaller SER than the CRBF, CMRAN and ASNN equalizers. It can be seen that the ASNN needs much more training symbols than CGAP-RBF.

3.3.5 Discussion

From the simulation results given above, the CGAP-RBF equalizer can achieve better performance than that of other equalizers, except Bayesian equalizer. Though Bayesian equalizer can get best performance and highest accuracy, it use all noise-free channel states as its centers, which is hard gotten in real

application. While for CGAP-RBF equalizer, it only needs a set pair of data for channel input and output, which can easily realize in real application by sampling. On the other hand, the complexity of Bayesian equalizer will exponentially increase with the increasing of channel order, which will result in time consumption. The CRBF, CMRAN and CGAP-RBF equalizers are all belongs to sequential learning algorithm. Furthermore, the CRBF equalizer is the fixed structure network using stochastic gradient criterion to adjust network free parameters. How to set the appropriate number of hidden neurons is a very common problem. The stochastic gradient criterion cannot guarantee convergence to globally optimum network parameters, which normally needs much training data to convergent to reasonable solutions and the learning speed is slow.

The CGAP-RBF and CMRAN equalizers are both self-constructing networks. They can achieve very compact network topology by growing and pruning criterion, which normally means a very fast learning speed. The CMRAN uses a sliding window in growing and pruning criterion. When the window size is set improperly, the hidden neurons will be added and deleted frequently, which will greatly affect the equalizer performance (refer to Fig. 3.22). However, because CGAP-RBF use a very simple formula (Eq. 3.12) to calculate the signification of hidden neurons, the progress of growing and pruning hidden neurons is very smooth by linking the learning accuracy to the significance of neurons. For network free parameter adjustment, CM-

3.3. Performance Evaluation of CGAP-RBF Equalizer

67

RAN change the parameters at the same time using extend Kalman filter (EKF) method. However, CGAP-RBF only adjusts the nearest neuron's parameters, which speeds up learning speed greatly. That is the reason why CGAP-RBF equalizer can obtain a very satisfied equalization performance compared with others.

In the following section, we will discuss equalizer complexity, input distribution and overlap factor (κ) in details.

Equalizer complexity

In this part, we will compare CGAP-RBF equalizer with CMRAN and CRBF equalizers in terms of training time and equalization time to evaluate their complexity. For the sake of brevity, only Example 3 of Cha and Kassam model [40] is considered for this comparison.

CGAP-RBF equalizer was trained with 1000 data symbols at 20dB of SNR. In order to obtain the same symbol error rate (SER), CRBF equalizer needed 13,500 data symbols and CMRAN needed 1400 data symbols, which are higher than the number of symbols required by CGAP-RBF.

Table 3.1 shows the comparison of CPU time of CGAP-RBF, CMRAN and CRBF equalizers for the training process of Example 3. The result is obtained using a P4/1.9GHZ personal computer. It can be seen that CGAP-RBF equalizer takes only 15.17s while CMRAN takes 65.7s and CRBF 98.7s.

3.3. Performance Evaluation of CGAP-RBF Equalizer68

Algorithms	# Neurons	# Data		CPU time	
		training	testing	training(sec)	testing(msec/sample)
CGAP-RBF	18	1000	10 ⁵	15.17	0.908
CMRAN	25	1400	10 ⁵	65.70	1.446
CRBF	30	13500	10 ⁵	98.78	1.722

Table 3.1: Equalizer complexity for Example 3

100 000 data symbols were used for testing purposes. Table 3.1 also shows the equalization time for the CGAP-RBF, CMRAN and CRBF equalizers for each test data. CGAP-RBF equalizer takes 0.908ms while CMRAN takes 1.446ms and CRBF 1.722ms per symbol training. Obviously, the CGAP-RBF needs much less training symbols, training time and testing time than CMRAN and CRBF.

Input distribution

In the above studies, CGAP-RBF equalizer used the formula (3.12) for significance based on uniform input distribution. This is because the expression for significance for uniform input distribution is computationally simpler compared to the general expression for significance given in formula (3.9). However, the equalizer inputs will always be non-uniform though the channel inputs are uniformly distributed. The equalizer input distribution in three

3.3. Performance Evaluation of CGAP-RBF Equalizer

69

problems has been checked and has been found to be highly non-uniform (Fig. 3.1). Though the distribution of equalizer input is non-uniform in the whole input space, the neighborhood distribution around each noise-free channel output state can be approximately looked as uniform. Furthermore, the small separated uniform regions are distributed around the origin, which satisfied the assumption condition in the derivation of uniform significance. Therefore, the CGAP-RBF equalizer derived with uniform input distribution performed better than all other equalizers even under non-uniform inputs.

Overlap factor κ

In order to discuss the effect of overlap factor κ to CGAP-RBF network, we rewrite the equation 3.12.

$$E_{sig}(k) = \frac{\|\alpha_k\|_2}{S(X)^{1/2}} \left(\frac{\pi^{3/2}}{\sqrt{2}} \sigma_k \right)^{m/2} \quad (3.27)$$

For new added neuron $K + 1$, the network parameters associated with the new neuron are selected as follows:

$$\begin{cases} \alpha_{K+1} = e_n \\ \mu_{K+1} = \mathbf{x}_n \\ \sigma_{K+1}^2 = \kappa(\mathbf{x}_n - \mu_{nr})^H(\mathbf{x}_n - \mu_{nr}) \end{cases} \quad (3.28)$$

Then, the significance for the new added hidden neuron can be calculated by:

$$E_{sig}(K + 1) = \frac{\|e_n\|_2}{S(X)^{1/2}} \left(\frac{\pi^{3/2}}{\sqrt{2}} \kappa(\mathbf{x}_n - \mu_{nr})^H(\mathbf{x}_n - \mu_{nr}) \right)^{m/2} \quad (3.29)$$

3.3. Performance Evaluation of CGAP-RBF Equalizer70

It can be seen that the increasing value of κ will result in more hidden neurons adding to the network, because the growing condition $E_{sig}(K + 1) > e_{goal}$ can be easily satisfied.

κ	# Neurons		Training RMSE		Testing RMSE		Training time		log10(SER)	
	Mean	Dev	Mean	Dev	Mean	Dev	Mean	Dev	Mean	Dev
0.1	5.9	1.21	0.42	0.03	0.43	0.04	1.57	0.38	-2.15	0.006
0.2	16.5	1.93	0.35	0.009	0.36	0.008	7.66	1.62	-3.73	0.022
0.3	27.4	1.76	0.32	0.008	0.33	0.008	23.5	3.86	-3.84	0.049
0.4	35.6	1.95	0.30	0.007	0.32	0.005	42.9	4.78	-4.10	0.059
0.5	41.6	2.28	0.29	0.01	0.31	0.008	61.66	9.05	-3.00	0.007
0.6	48.6	2.76	0.29	0.01	0.31	0.01	89.48	9.85	-2.56	0.011
0.7	56.1	1.92	0.29	0.006	0.31	0.009	114	7.61	-2.80	0.007

Table 3.2: Performance comparison of CGAP-RBF equalizer at different κ (Example 3).

Table 3.2 is the performance comparison of CGAP-RBF equalizer at different κ for Example 3. The results are obtained after 20 independent runs in one P4/2.8GHZ personal computer. It can be seen that the number of hidden neurons always increase with increment of κ . However, symbol error rate (SER) deduces with the increasing of κ before $\kappa = 0.4$ and increase again after $\kappa = 0.4$. This is because the over fitting phenomenon occurs with the increasing of hidden neurons. Therefore, the best value of κ for CGAP-RBF is 0.2 – 0.4 and $\kappa = 0.2$ is chosen in our simulations.

3.4 Conclusions

A complex-valued growing and pruning RBF network (CGAP-RBF) is presented for channel equalization of 4-QAM and 16-QAM signals in this chapter. CGAP-RBF learning algorithm uses the notion of significance for the growth and pruning of hidden neurons. CGAP-RBF performance evaluation has been carried out based on three channel equalization problems of increasing complexity. Results indicate superior performance of CGAP-RBF equalizer in terms of SER, complexity and learning speed.

Recently, Kim and Adali [63, 37] have proposed a fully complex MLP equalizer in which the activation functions of the neural network are all complex functions unlike the other schemes mentioned above. They have shown the performance of their MLP to be superior to conventional split complex MLP equalizers. It would be interesting to extend this to complex RBF equalizers.

Chapter 4

Complex Incremental Extreme Learning Machine

Neural networks as a kind of important artificial intelligence method have attracted considerable attention, which have been successfully applied to many application fields due to their approximation capability. Recently, with a surge of interest in wireless and mobile communication applications, the theoretical analysis of complex domain is highly demanded in those applications. Two challenges exist in designing a neural network on complex domain. One is to find a proper nonlinear complex activation function. The main reason is *“the conflict between the boundedness and the differentiability of a nonlinear complex function in the complex domain”* [42]. The other is how to shorten cost time in the training process of neural networks as soon as possible. Re-

cently an incremental algorithm was proposed by Huang et al. [43](I-ELM), which randomly chooses hidden nodes and then analytically determines the output weights connecting the hidden layer and the output layer. Though hidden nodes are generated randomly, the network constructed by I-ELM is still universal approximation. Simulation results have also proved that it is a fast and efficient method for real world regression problems. In this paper, we extend I-ELM into the applications of complex domain. Complex feedforward neural networks with randomly chosen hidden nodes and constructed by this algorithm can work as universal approximators. Due to the random character, the corresponding neural networks can provide good generalization performance at extremely fast learning speed. Some function approximation cases further verify our conclusion.

4.1 Introduction

Single-hidden layer feedforward neural networks (SLFNs) have recently attracted an extensive research interest in many development and application fields due to their approximation capability[106, 107, 108, 109, 110]. So far, many researches show that SLFNs can accurately describe nonlinear mappings with a finite number of neurons. Based on above approximation theories, many evolutionary computation algorithms and optimization analysis of neural networks have been developed such that neural networks can achieve

better generalization performance.

Recently, a novel learning method named Extreme Learning Machine (ELM) has been proposed by Huang, et al. [44, 45, 111, 112, 113]. In ELM, the hidden nodes can be randomly chosen (the hidden node parameters are generated based on some continuous distribution probabilities). Then the output layer weights are analytically determined by using a least-squares method and the corresponding networks by the algorithm can learn N distinct observations with arbitrarily small error. Since ELM need not adjust hidden node parameters, ELM runs extremely fast. Though a large number of experimental results on ELM show that the parameters of networks need not be adjusted at all, it still remains unknown whether neural networks with random parameters are universal approximators. Huang et. al has proved the universal approximation capability of ELM in an incremental method (IELM), which rigorously proved that single-hidden layer feedforward networks (SLFNs) with random hidden nodes can work as universal approximators by only calculating the output weights β_i linking the hidden layer to the output nodes.

Neural networks have been used in complex fields such as wireless and mobile communications applications [40, 41, 64]. However it still lacks of rigorous research. The main reason is the challenges in finding a proper non-linear complex activation function on complex fields to construct a neural network to process complex signal [42, 63, 114]. According to complex func-

tion analysis, the main difficulty is the conflict between the boundedness and the differentiability of complex function in the entire complex plane. In fact, a bounded entire function must be a constant in the complex plane, where the entire function means that the function is analytic every point in the complex plane. If the function is not analytic, there must be a singularity point at least, i.e., removable, isolated and essential singularities. Thus, it is difficult for researchers to design a neural network in complex fields. Recently, Kim and Adali [42] proved the approximation ability of multilayer feedforward neural networks on the complex domain. Based on the different characteristics of complex activation function, three results about the approximation capability of the fully complex multilayer feedforward neural networks are provided in [42].

In this chapter, we further enhance the universal approximation capability of complex feedforward neural networks. By extending I-ELM into complex domain, we rigorously prove that complex feedforward neural networks with randomly chosen hidden nodes and with complex I-ELM can work as universal approximators. Two complex-valued function approximation problems are used to evaluate the validity of our proof. The simulation results show that the complex-valued incremental ELM (C-IELM) network has universal approximation capability in complex domain at extremely fast learning speed.

4.2 Preliminaries

4.2.1 Review of I-ELM in real domain

In this section, we first introduce I-ELM with randomly generated additive and RBF nodes in Huang's earlier work.

Without loss of generality, we assume that the network has only one linear output neuron. It can be easily extended all the analysis into multi nonlinear output neurons cases. A standard SLFNs functions with n hidden neurons can be represented by

$$f_n(\mathbf{x}) = \sum_{i=1}^n \beta_i g_i(\mathbf{x}), \mathbf{x} \in \mathbf{R}^d, \beta_i \in R \quad (4.1)$$

where $g_i(\mathbf{x}) = g(\mathbf{a}_i, b_i, \mathbf{x})$ denote the i -th hidden node activation function, $\mathbf{a}_i$ and b_i are the learning parameters of hidden nodes and β_i is the weight connecting the i -th hidden node to the output node. Based on different parameters' combinations, there are mainly two neural networks models, i.e., additive model and RBF model. For the additive model, the activation function $g(x) : R \rightarrow R$ takes the form by

$$g(\mathbf{a}_i, b_i, \mathbf{x}) = g(\mathbf{a}_i \cdot \mathbf{x} + b_i), \mathbf{a}_i \in \mathbf{R}^d, b_i \in R \quad (4.2)$$

where $\mathbf{a}_i$ is the weight vector connecting the input layer to the i -th hidden node and b_i is the bias of the i -th hidden node. $\mathbf{a}_i \cdot \mathbf{x}$ denotes the inner product of vectors $\mathbf{a}_i$ and $\mathbf{x}$ in R^d . For the RBF model, the activation function

$g(x) : R \rightarrow R$ takes the form by

$$g(\mathbf{a}_i, b_i, \mathbf{x}) = g\left(\frac{\|\mathbf{x} - \mathbf{a}_i\|}{b_i}\right), \mathbf{a}_i \in \mathbf{R}^d, b_i \in R^+ \quad (4.3)$$

where $\mathbf{a}_i$ is the center of the i -th RBF node and b_i is the impact of the i -th RBF node. R^+ indicates the set of all positive real value.

Definition 4.2.1. {P.334 of [115]} *A function $g(x) : R \rightarrow R$ is said to be piecewise continuous if it has only a finite number of discontinuities in any interval, and its left and right limits are defined (not necessarily equal) at each discontinuity.*

Let $e_n \equiv f - f_n$ denote the residual error function for the current network f_n with n hidden neurons where $f \in L^2(X)$ is the target function. In Huang's paper[43], the form of I-ELM is:

$$f_n(\mathbf{x}) = f_{n-1}(\mathbf{x}) + \beta_n g_n(\mathbf{x}) \quad (4.4)$$

which means that for the new node g_n , we randomly choose its parameters and analytically calculate the corresponding output weight β_n regardless of previous nodes. Unlike others traditional algorithms, which usually find proper parameter based on some optimization techniques such that $\lim_{n \rightarrow \infty} \|f - f_n\| = 0$, neural networks with I-ELM are still universal approximation though parameters are chosen randomly. The corresponding theorem is following:

Lemma 4.2.1. [43] *Given any bounded nonconstant piecewise continuous function $g : \mathbf{R} \rightarrow \mathbf{R}$ for additive nodes or any integrable piecewise con-*

tinuous function $g : \mathbf{R} \rightarrow \mathbf{R}$ and $\int_{\mathbf{R}} g(x)dx \neq 0$ for RBF nodes, for any continuous target function f and any randomly generated function sequence $\{g_n\}$, $\lim_{n \rightarrow \infty} \|f - f_n\| = 0$ holds with probability one if

$$\beta_n = \frac{\langle e_{n-1}, g_n \rangle}{\|g_n\|^2} \quad (4.5)$$

Remark: Lemma 4.2.1 shows that for new added neuron, I-ELM is freezing previous existing neurons, and the new node needn't be tune. The major contribution of I-ELM is that it breaks through the restriction of others traditional theories, and shows that neural networks with random nodes may be universal approximation. I-ELM covers a broad class of functions: the activation functions for additive nodes can be any bounded nonconstant piecewise continuous functions $g : R \rightarrow R$ and the activation functions for RBF nodes can be any integrable piecewise continuous functions $g : R \rightarrow R$ and $\int_R g(x)dx \neq 0$. I-ELM is not only efficient for SLFNs with continuous (including non-differentiable) activation functions but also for SLFNs with piecewise continuous (such as threshold) activation functions.

4.2.2 Symbols and Theorems in complex domain

Without loss of generality, assume that the network has only one linear output neuron. It can be easily seen that the extension of all the analysis conducted in this paper to multi nonlinear output neurons cases is straightforward. The output of neural networks with n hidden neurons can be rep-

resented by

$$f_n(\mathbf{z}) = \sum_{i=1}^n \beta_i g_i(\mathbf{z}) \quad (4.6)$$

where β_i the weight connecting the i -th hidden neuron to the output neuron, and $g_i(\mathbf{z})$ is the activation function of the i -th hidden neuron for an input vector $\mathbf{z} \in \mathbb{C}^d$:

$$g_i(\mathbf{z}) = g(\mathbf{w}_i \cdot \mathbf{z} + b_i) \quad (4.7)$$

where $\mathbf{w}_i \in \mathbb{C}^d$ and $b_i \in \mathbb{C}$, $i = 1, \dots, n$, are weights and bias. We assume that $g(\mathbf{z})$ is nonlinear bounded integrable and continuous in bounded space Z .

Definition 4.2.2. {P.25 of [116]} *A function is called an inner product: let $\mathcal{H}$ be a linear space over $\mathbb{C}$ with a given complex-valued function of two variables $\langle x, y \rangle : \mathcal{H} \times \mathcal{H} \mapsto \mathbb{C}$, which has the following properties:*

1) *linearity with respect to the first argument:*

$$\langle ax_1 + bx_2, y \rangle = a \langle x_1, y \rangle + b \langle x_2, y \rangle; \quad (4.8)$$

2) $\overline{\langle x, y \rangle} = \langle y, x \rangle$; *this implies what is sometimes called semi-linearity with respect the second argument: $\langle x, ay_1 + by_2 \rangle = \bar{a} \langle x, y_1 \rangle + \bar{b} \langle x, y_2 \rangle$.*

3) *non-negativeness: $\langle x, x \rangle \geq 0$, $\langle x, x \rangle = 0$ if and only if $x = 0$.*

Let $L^2(Z)$ be a space of functions f on a measurable compact subset Z in the d -dimensional space $\mathbb{C}^d$ such that $|f|^2$ are integrable. For $u, v \in L^2(Z)$,

the inner product $\langle u, v \rangle$ is defined by

$$\langle u, v \rangle = \int_Z u(\mathbf{z}) \overline{v(\mathbf{z})} d\mathbf{z} \quad (4.9)$$

The norm in L^2 space will be denoted as $\| \cdot \|$, and the closeness between network function f_n and the target function f is measured by the L^2 distance:

$$\|f_n - f\| = \left[\int_Z (f_n(\mathbf{z}) - f(\mathbf{z})) \overline{(f_n(\mathbf{z}) - f(\mathbf{z}))} d\mathbf{z} \right]^{1/2} \quad (4.10)$$

In this paper the sample input space Z is always considered as a bounded measurable compact subset of the space $\mathbf{C}^d$. It is very unlikely that one will deal with non-measurable sample input sets in the applications of neural networks.

Similar, we define piecewise continuous function on complex domain:

Definition 4.2.3. A function $g(x) : \mathcal{C} \rightarrow \mathcal{C}$ is said to be piecewise continuous if it has only a finite number of discontinuities in any interval.

Definition 4.2.4. The function sequence $\{g_n = g(\mathbf{a}_n \cdot \mathbf{z} + b_n)\}$ is said randomly generated if the corresponding parameters $(\mathbf{a}_n, b_n)$ are randomly generated from $\mathcal{C}^d \times \mathcal{C}$ based on a continuous sampling distribution probability.

Remark 1: As done in our simulations, one may randomly generate sequence $\{g_n\}$ based on a uniform sampling distribution probability.

Definition 4.2.5. A node is called a random node if its parameters $(\mathbf{a}, b)$ are randomly generated based on a continuous sampling distribution probability.

Definition 4.2.6. {P.1656 of [42]} *A function σ is discriminating if for a measure $\mu \in M(\mathcal{I}_n)$, $\int_{\mathcal{I}_n} \sigma(\mathbf{w}^T \cdot \mathbf{z} + \theta) = 0$ for all $\mathbf{w} \in \mathbb{C}^n$ and $\theta \in \mathbb{C}$, implies that $\mu = 0$ (a.e. for a measurable function σ . In other words, σ is discriminating if it does not vanish anywhere in the domain $\mathcal{I}_n$, where $\mathcal{I}_n$ denotes the n -dimensional complex unit cube, $[0, 1]^n$).*

4.2.3 Necessary Lemmas

Some lemmas which are used to prove our main theorem are provided in this section.

Lemma 4.2.1. {P.81 of [117]} *The space of L^2 is complete.*

Lemma 4.2.2. {Theorem 1 of [42]} *Let $\sigma : \mathbb{C} \rightarrow \mathbb{C}$ be any complex continuous discriminatory function. Let $\mathcal{I}_d$ denote the d -dimensional complex unit cube $[0, 1]^d$. Then the finite sums of the product of the form $f_n(\mathbf{z}) = \sum_{i=1}^n \beta_i \prod_{l=1}^{s_i} \sigma(\mathbf{a}_{il} \cdot \mathbf{z} + b_i)$ are dense in $C(\mathcal{I}_d)$, that is, $\forall f \in C(\mathcal{I}_d)$ and $\varepsilon > 0$, $\exists f_n(\mathbf{z})$ such that $|f_n(\mathbf{z}) - f(\mathbf{z})| < \varepsilon$, $\forall \mathbf{z} \in \mathcal{I}_d$, where $\mathbf{a}_{il} \in \mathbb{C}^d$ and $b_i \in \mathbb{C}$.*

Lemma 4.2.2 shows that if σ is complex continuous discriminatory, for any target complex continuous function f there exists f_n such as f_n converges to f everywhere in the bounded set $\mathcal{I}_d$, thus we further have $\|f_n(\mathbf{z}) - f(\mathbf{z})\| < \varepsilon$ which is weaker than $|f_n(\mathbf{z}) - f(\mathbf{z})| < \varepsilon$. Therefore, we have

Lemma 4.2.3. *Given any complex continuous discriminatory function $\sigma : C \rightarrow C$, for any target continuous function f and $\varepsilon > 0$ there exists f_n such that $\|f_n(\mathbf{z}) - f(\mathbf{z})\| = \|\sum_{i=1}^n \beta_i \prod_{l=1}^{s_i} \sigma(\mathbf{a}_{il} \cdot \mathbf{z} + b_i) - f\| < \varepsilon$, where $\mathbf{a}_{il} \in \mathbf{C}^d$ and $b_i \in C$.*

Lemma 4.2.4. {Theorem 2 of [42]} *Let $\sigma : C \rightarrow C$ be any complex bounded measurable discriminatory function. Then the finite sums of the form $f_n(\mathbf{z}) = \sum_{i=1}^n \beta_i \prod_{l=1}^{s_i} \sigma(\mathbf{a}_{il} \cdot \mathbf{z} + b_i)$ are dense in $L^1(\mathcal{I}_d)$, where $\mathbf{a}_{il} \in \mathbf{C}^d$ and $b_i \in C$.*

Lemma 4.2.3 shows the case where the activation function σ is complex continuous discriminatory, however in this case the activation function σ may not be bounded. Lemma 4.2.4 shows the case where the activation function σ is bounded but may not be continuous. As the supremum norm in $L^1(\mu)$ can be generalized to $L^p(\mu)$ -norm with $0 < p < \infty$, σ may be piecewise continuous, we further have

Lemma 4.2.5. *Given any complex bounded nonlinear piecewise continuous function $\sigma : C \rightarrow C$, $f_n(\mathbf{z}) = \sum_{i=1}^n \beta_i \prod_{l=1}^{s_i} \sigma(\mathbf{a}_{il} \cdot \mathbf{z} + b_i)$ are dense in $L^2(Z)$, where $\mathbf{a}_{il} \in \mathbf{C}^d$ and $b_i \in C$.*

4.3 Approximation

In this section we can show that any continuous target function $f : \mathbf{C}^d \rightarrow C$ can be approximated with any arbitrarily small error by an incremental fully

complex ELM where the complex hidden nodes are randomly added one by one and will be fixed once added. In fact, given any complex continuous discriminatory or any complex bounded nonlinear piecewise continuous function $\sigma : C \rightarrow C$, and any randomly generated function sequence $\{g_i(\mathbf{z})\}$:

$$g_i(\mathbf{z}) = \prod_{l=1}^{s_i} \sigma(\mathbf{a}_{il} \cdot \mathbf{z} + b_i) \quad (4.11)$$

where $\mathbf{a}_{il}$ and b_i are randomly generated fully independently from the target function f based on any continuous distribution probability, then for any small positive value ϵ , there exists a network sequence $\{f_n\}$, we have $\lim_{n \rightarrow \infty} \|f_n - f\| = 0$ if $\beta_n = \frac{\langle e_{n-1}, g_n \rangle}{\|g_n\|^2}$.

Theorem 4.3.1. *Given any complex continuous discriminatory or any complex bounded nonlinear piecewise continuous function $\sigma : C \rightarrow C$, for any target complex continuous function $f : \mathbf{C}^d \rightarrow C$ and any randomly generated function sequence $\{g_n = \prod_{l=1}^{s_n} \sigma(\mathbf{a}_{nl} \cdot \mathbf{z} + b_n)\}$, $\lim_{n \rightarrow \infty} \|f - f_n\| = 0$ holds with probability one if*

$$\beta_n = \frac{\langle e_{n-1}, g_n \rangle}{\|g_n\|^2}. \quad (4.12)$$

Proof. Since complex space is also measurable geometry space, therefore the whole proof is similar to [43]{pp. 881-884}. The main difference is that we only need to migrate the inner product in the whole proof from the real domain to complex domain.

Since σ is a complex continuous discriminatory or complex bounded nonlinear piecewise continuous function, $g_i(\mathbf{z}) = \prod_{l=1}^{s_i} \sigma(\mathbf{a}_{il} \cdot \mathbf{z} + b_i) \in L^2(Z)$

4.3. Approximation

84

and $\|g_n\| = \int_Z g_n \cdot \overline{g_n} d\mathbf{z} \neq 0$. The target function f is continuous, we have $f \in L^2(Z)$. According to Lemma 4.2.1, $e_n = f - f_n \in L^2(Z)$. Let $\Delta = \|e_{n-1}\|^2 - \|e_n\|^2$, then we have

$$\begin{aligned}
 \Delta &= \|e_{n-1}\|^2 - \|e_n\|^2 \\
 &= \langle e_{n-1}, e_{n-1} \rangle - \langle e_{n-1} - \beta_n g_n, e_{n-1} - \beta_n g_n \rangle \\
 &= \langle e_{n-1}, e_{n-1} \rangle - (\langle e_{n-1}, e_{n-1} \rangle - \langle e_{n-1}, \beta_n g_n \rangle - \langle \beta_n g_n, e_{n-1} \rangle + \langle \beta_n g_n, \beta_n g_n \rangle) \\
 &= \overline{\beta_n} \langle e_{n-1}, g_n \rangle + \beta_n \langle g_n, e_{n-1} \rangle - \beta_n \overline{\beta_n} \langle g_n, g_n \rangle \\
 &= \overline{\beta_n} \langle e_{n-1}, g_n \rangle + \beta_n \overline{\langle e_{n-1}, g_n \rangle} - \beta_n \overline{\beta_n} \langle g_n, g_n \rangle \\
 &= \|g_n\|^2 \left(\frac{\langle e_{n-1}, g_n \rangle \overline{\langle e_{n-1}, g_n \rangle}}{\|g_n\|^4} - \left(\beta_n - \frac{\langle e_{n-1}, g_n \rangle}{\|g_n\|^2} \right) \left(\overline{\beta_n - \frac{\langle e_{n-1}, g_n \rangle}{\|g_n\|^2}} \right) \right)
 \end{aligned} \tag{4.13}$$

Δ is maximized iff $\beta_n = \frac{\langle e_{n-1}, g_n \rangle}{\|g_n\|^2}$, meaning that $\|e_n\| = \|f - (f_{n-1} + \beta_n g_n)\|$ achieves its minimum iff $\beta = \beta_n = \frac{\langle e_{n-1}, g_n \rangle}{\|g_n\|^2}$. The result is consistent with the real domain case.

With Lemma 4.2.3 and Lemma 4.2.5 we can prove $\|e_n\|$ converges to zero in the same proof method given in [43]{pp. 881-884}. For the sake of brevity, readers can refer to [43] for details as it does not convey any new idea to repeat the same proof procedure. $\square$

Remark: Since continuous functions on any compact subset are bounded, activation functions in Theorem 4.3.1 are continuous functions or bounded non-constant piecewise continuous functions. For bounded nonconstant piecewise continuous functions, those discontinuous points are also singularity points.

In complex analysis, three types of singularities are known: removable, isolated and essential. For isolated and essential functions, they both contain unbounded cases, so we do not consider the two functions. For removable functions, $f(z)$ has single point singularity at z_0 , the singularity is said to be removable if $\lim_{z \rightarrow z_0} f(z)$ exists, i.e., bounded.

Remark: In Theorem 4.3.1, we successfully prove that complex neural networks with the incremental algorithm and with arbitrary weights and bias can own universal approximation capability. We can choose arbitrary weights and bias to construct the sequence $\{g_n\}_{n=1}^{\infty}$. Without calculating weight and bias to maximize $\frac{\langle e_{n-1}, g \rangle \overline{\langle e_{n-1}, g \rangle}}{\|g\|^2}$, the corresponding networks can respond to online data for real-time applications.

4.4 Experimental Verification

In the above section, we provide a theoretical justification for the incremental feedforward networks in complex domain. In this section, simulation results are given to verify the theory, and it further demonstrates the universal approximation capability of complex neural networks.

4.4.1 Incremental Algorithm

In this section, we will introduce a detailed algorithm for arbitrary weights and bias. Based on Theorem 4.3.1, we know that the weights β_n should be chosen as $\frac{\langle e_{n-1}, g_n \rangle}{\|g_n\|^2}$ for new adding neurons. According to the definition of Hermitian inner product, we have $\langle u, v \rangle = \int_X u(\mathbf{z}) \overline{v(\mathbf{z})} d\mathbf{z} = \sum_{p=1}^N u(\mathbf{z}_p) \overline{v(\mathbf{z}_p)}$, so an estimate β_n based on the training samples is

$$\beta_n = \frac{E_{n-1} \cdot H^*}{H \cdot H^*} = \frac{\sum_{p=1}^N e_{n-1}(p) \overline{g_n(p)}}{\sum_{p=1}^N g_n(p) \overline{g_n(p)}} \quad (4.14)$$

where H^* means complex conjugate transposition, $g_n(p)$ is the n -th kernel function in complex neural networks for the input of p -th training sample and $e(p)$ is the corresponding residual error before this new hidden neuron is added. $H = [g_n(1), \dots, g_n(N)]^T$ is the activation vector of the new neuron for all the N training samples and $E_{n-1} = [e_{n-1}(1), \dots, e_{n-1}(N)]^T$ is the residual vector before this new hidden neuron added. In real applications, one may not really wish to get zero approximation error by adding infinite neurons to the network, a maximum number of hidden neurons is normally given. The detail algorithm is summarized as follows:

Algorithm: Given a training set $\aleph = \{(\mathbf{z}_i, t_i) | \mathbf{z}_i \in \mathbf{C}^d, t_i \in C, i = 1, \dots, N\}$, activation function $g(x)$, maximum neuron number $\tilde{N}_{\max}$ and expected learning accuracy ϵ ,

step 1 Initialization: Let $\tilde{N} = 0$ and residual error $E = t$, where $t = [t_1, \dots, t_N]^T$.

4.4. Experimental Verification

87

step 2 **Learning step:**

while $\tilde{N} < \tilde{N}_{\max}$ and $\|E\| > \epsilon$

- (a) Increase by 1 the number of hidden neurons $\tilde{N}$: $\tilde{N} = \tilde{N} + 1$.
- (b) Assign arbitrary center $\mathbf{w}_{\tilde{N}}$ and width $b_{\tilde{N}}$ for new hidden neuron $\tilde{N}$.
- (c) Calculate the output weight $\beta_{\tilde{N}}$ for the new hidden neuron:

$$\beta_{\tilde{N}} = \frac{E \cdot H_{\tilde{N}}^*}{H_{\tilde{N}} \cdot H_{\tilde{N}}^*} \quad (4.15)$$

- (d) Calculate the residual error after adding the new hidden neuron $\tilde{N}$:

$$E = E - \beta_{\tilde{N}} \cdot H_{\tilde{N}} \quad (4.16)$$

endwhile

In *step 1*, since there is no neuron in the network, the residual error is initially set as the expected target vector of the training data set. Learning will stop when the number $\tilde{N}$ of hidden neurons has exceeded the predefined maximum number $\tilde{N}_{\max}$ or the residual error E becomes less than the expected one. The E in the right side of equation (4.16) represents the residual error vector before the new neuron added and the E in the left side represents the residual error vector after the new neuron added, which is consistent to $E_{\tilde{N}} = f - f_{\tilde{N}} = E_{\tilde{N}-1} - \beta_{\tilde{N}} g_{\tilde{N}}$.

4.4.2 Simulation

In this section, two approximation problems in complex domain used in [62] are performed to show that the proposed theory is correct. In all the simulations, 10000 training samples and 1000 testing samples are randomly drawn from the interval $[0 + i0, 1 + i]$. Both the input weight vectors $\mathbf{w}_k$ and biases b_k of the complex I-ELM are randomly chosen from a complex area centered at the origin with the radius set as 1. $\tilde{N}_{\max}$ is set 6000 and $\epsilon = 0.01$. The simulation results are obtained after 10 independent run for each method using a P4/2.8GHZ personal computer.

As mentioned in Lemma 4.2.1, the function which is satisfied in theory proof needs to be bounded nonconstant piecewise continuous function. Though many fully complex activation functions have been proposed by Kim and Adali [42], only the functions which have removable singularity can satisfy the condition. They are the functions: $\arcsin(z) = \int_0^z \frac{dt}{(1-t^2)^{1/2}}$, $\arccos(z) = \int_0^z \frac{dt}{(1-t^2)^{1/2}}$, and $\operatorname{arcsinh}(z) = \int_0^z \frac{dt}{(1+t^2)^{1/2}}$, where $z \in C$.

Example 5: The non-analytic function used in [62] is given by:

$$f(z) = f(x + iy) = e^{iy}(1 - x^2 - y^2) \quad (4.17)$$

Fig. 4.1 are the average testing root mean square error (RMSE) with different activation functions during the training process. It is obviously seen that the learning convergence curves decrease with the increasing of neuron

Figure 4.1: Learning convergence for arcsin, arcsinh and arccos functions (Example 5)

Figure 4.2: Training time for different functions (Example 5)

number, which indicates the correctness of the proposed theory. Fig. 4.2 are the spent learning time curves for different functions. From Fig. 4.2, it shows that training time is linearly increasing with the learning step, which

4.4. Experimental Verification

90

is consistent with the analysis on real domain.

Example 6: Here, we will use one analytic function to evaluate the approximation capability of different functions. The analytic function is given by:

$$f(z) = f(x + iy) = \sin(x)\cosh(y) + i\cos(x)\sinh(y) \quad (4.18)$$

Fig. 4.3 are the average testing RMSE, which is decreasing with the increase of hidden nodes. Fig. 4.4 are the spent learning time curves for different functions, which is linearly increasing with the learning step. The two examples both verify our theory, i.e., feedforward neural networks with random complex nodes can work as universal approximators.

Figure 4.3: Learning convergence for arcsin, arcsinh and arccos functions (Example 6)

Figure 4.4: Training time for different functions (Example 6)

4.4.3 Discussion

In this chapter, the approximation capability of I-ELM in complex domain is theoretically proved. Two function approximation problems are performed to demonstrate its correctness. From theory point, our theorem breaks through all the previous theory about approximation capability of neural networks on complex domain, and points out that feedforward neural networks with random complex nodes can work as universal approximators. From the simulation results, our theorem is further verified. The average testing RMSE is convergent and decreasing with the increasing of hidden neurons. When the hidden neurons are sufficiently enough, the learning error will converge to zero. The functions ($\arcsin$, $\operatorname{arcsinh}$ and $\arccos$) used in our simulations are satisfied the condition of bounded nonconstant piecewise continuous function

4.4. Experimental Verification

92

and is suitable to use as activation function in complex domain approximation.

Chapter 5

Fully Complex Extreme Learning Machine

Recently, a new learning algorithm for the feedforward neural network named the extreme learning machine (ELM) has been proposed by Huang, et al. [44, 45], which can give better performance than traditional tuning-based learning methods for feedforward neural networks in terms of generalization and learning speed. In this chapter, we first extend the ELM algorithm from the real domain to the complex domain, and then apply the fully complex extreme learning machine (C-ELM) for nonlinear channel equalization applications. The simulation results show that the ELM equalizer significantly outperforms other neural network equalizers such as the complex minimal re-

⁰The main idea of this chapter can be found in “Fully Complex Extreme Learning Machine,” *Neurocomputing*, Vol. 68, pp. 306-314, 2005.

source allocation network (CMRAN), complex radial basis function (CRBF) network and complex backpropagation (CBP) equalizers. C-ELM achieves much lower symbol error rate (SER) and has faster learning speed.

5.1 Introduction

In high speed digital communication systems, equalizers are used very often at receivers to recover the original symbols from the received signals. Real-valued neural network models such as feedforward neural networks, radial basis function (RBF) networks and recurrent neural networks have been successfully used for solving equalization problems as neural networks are well suited for non-linear classification problems [28]. Complex-valued neural networks have attracted considerable attention in channel equalization applications in the past decade. Cha and Kassam [40] have proposed a Complex-valued Radial Basis Function (CRBF) network which adopts the stochastic gradient learning algorithm to adjust parameters. Compared with previously existing equalizers, the CRBF equalizer is superior in terms of symbol error rate (SER). Jianping et al. [41] have developed a Complex-valued Minimal Resource Allocation Network (CMRAN) equalizer. Applying the growing and pruning criterion, the CMRAN equalizer realizes a more compact structure and obtains better performance than CRBF and many other equalizers. However, it should be noted that although the inputs and

the centers of CRBF and CMRAN are complex-valued, the basis functions still remain real-valued. In fact, as pointed out by Kim and Adali [42], split-complex activation (basis) functions consisting of two real-valued activation functions, one processing the real part and the other processing the imaginary part, have been traditionally employed in these complex-valued neural networks. Kim and Adali [37, 42] have proposed an important complex neural network model - a fully complex multilayer perceptron (MLP) which uses true complex-valued activation function. It has been rigorously proved [42] that with very mild condition on the complex activation functions the fully complex MLPs can universally approximate any continuous complex mappings. The corresponding fully complex backpropagation (CBP) learning algorithm with fully complex activation function has also been successfully used in communication applications [37].

Recently, a new learning algorithm for Single-hidden-Layer Feedforward Neural network (SLFN) named the extreme learning machine (ELM) has been proposed by Huang, et al. [44, 45]. Unlike traditional approaches (such as BP algorithms) which may face difficulties in manually tuning control parameters (learning rate, learning epochs, etc) and/or local minima, ELM avoids such issues and reaches good solutions analytically. The learning speed of ELM is extremely fast compared to other traditional methods. In this chapter, we first extend the ELM algorithm from the real domain to the complex domain where the fully complex activation functions introduced

5.2. Complex Extreme Learning Machine (C-ELM) Algorithm 96

by Kim and Adali [42] are used. Similar to ELM, the input weights (linking the input layer to the hidden layer) and hidden layer biases of C-ELM are randomly chosen based on some continuous distribution probability (such as uniform distribution probability used in our simulations) and the output weights (linking the hidden layer to the output layer) are then analytically calculated. The C-ELM is used for equalization of a complex nonlinear channel with QAM signals. Simulation results show that the C-ELM equalizer is superior to CRBF [40], CMRAN [41] and CBP [37] equalizers in terms of symbol error rate (SER) and learning speed. C-ELM also avoids local minima and tuning control parameters (learning rate, learning epochs, etc).

5.2 Complex Extreme Learning Machine (C-ELM) Algorithm

Given a series of complex-valued training samples $(\mathbf{x}_t, \mathbf{y}_t)$, $t = 1, 2, \dots, N$, where $\mathbf{x}_t \in \mathbb{C}^n$ and $\mathbf{y}_t \in \mathbb{C}^m$, the actual outputs of the single-hidden-layer feedforward network (SLFN) with complex activation function $g_c(z)$ for these N training data is given by

$$\sum_{k=1}^K \beta_k g_c(\mathbf{w}_k \cdot \mathbf{x}_t + b_k) = \mathbf{o}_t, \quad t = 1, \dots, N, \quad (5.1)$$

where column vector $\mathbf{w}_k \in \mathbb{C}^n$ is the complex input weight vector connecting the input layer neurons to the k -th hidden neuron, $\beta_k = [\beta_{k1}, \beta_{k2}, \dots, \beta_{km}]^T \in$

5.2. Complex Extreme Learning Machine (C-ELM) Algorithm 97

$\mathbf{C}^m$ the complex output weight vector connecting the k -th hidden neuron and the output neurons, and $b_k \in \mathbb{C}$ is the complex bias of the k -th hidden neuron. $\mathbf{w}_k \cdot \mathbf{x}_t$ denotes the inner product of column vectors $\mathbf{w}_k$ and $\mathbf{x}_t$. g_c is a fully complex activation function.

The above N equations can be written compactly as

$$\mathbf{H}\beta = \mathbf{O} \quad (5.2)$$

and in practical applications the number K of the hidden neurons is usually much less than the number N of training samples and $\mathbf{H}\beta \neq \mathbf{Y}$, where

$$\begin{aligned} & \mathbf{H}(\mathbf{w}_1, \dots, \mathbf{w}_K, \mathbf{x}_1, \dots, \mathbf{x}_K, b_1, \dots, b_K) \\ &= \begin{bmatrix} g_c(\mathbf{w}_1 \cdot \mathbf{x}_1 + b_1) & \cdots & g_c(\mathbf{w}_K \cdot \mathbf{x}_1 + b_K) \\ \vdots & \cdots & \vdots \\ g_c(\mathbf{w}_1 \cdot \mathbf{x}_N + b_1) & \cdots & g_c(\mathbf{w}_K \cdot \mathbf{x}_N + b_K) \end{bmatrix}_{N \times K} \end{aligned} \quad (5.3)$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_K^T \end{bmatrix}_{K \times m}, \quad \mathbf{O} = \begin{bmatrix} \mathbf{o}_1^T \\ \vdots \\ \mathbf{o}_N^T \end{bmatrix}_{N \times m} \quad \text{and} \quad \mathbf{Y} = \begin{bmatrix} \mathbf{y}_1^T \\ \vdots \\ \mathbf{y}_N^T \end{bmatrix}_{N \times m} \quad (5.4)$$

The complex matrix $\mathbf{H}$ is called the hidden layer output matrix. Using the analysis similar to that of ELM [44, 45] and using the proof given in ([118] p.252 and [119] Theorem 2.1) we can easily show that the input weights $\mathbf{w}_k$ and hidden layer biases b_k of the SLFNs with complex activation functions (which are infinitely differentiable) can be randomly chosen and fixed based

5.2. Complex Extreme Learning Machine (C-ELM) Algorithm 98

on some continuous distribution probability instead of been trivially tuned.¹ As analyzed by Huang et. al. [45, 44] for fixed input weights $\mathbf{w}_k$ and hidden layer biases b_k we can get the *least-squares solution* $\hat{\beta}$ of the linear system $\mathbf{H}\beta = \mathbf{Y}$ with *minimum norm* of output weights β , which usually tend to have good generalization performance: (Refer to Huang et. al. [113, 111, 45, 44] for detailed analysis.)

The resulting $\hat{\beta}$ is given by:

$$\hat{\beta} = \mathbf{H}^\dagger \mathbf{Y} \quad (5.5)$$

where complex matrix $\mathbf{H}^\dagger$ is the *Moore-Penrose generalized inverse* (pp. 163-169 of [120]) of complex matrix $\mathbf{H}$. Thus, ELM can be extended from the real domain to a fully complex domain in a straightforward manner. The three steps in the fully complex ELM (C-ELM) algorithm can be summarized as:

Algorithm C-ELM: Given a training set $\aleph = \{(\mathbf{x}_t, \mathbf{y}_t) | \mathbf{x}_t \in \mathbf{C}^n, \mathbf{y}_t \in \mathbf{C}^m, t = 1, \dots, N\}$, complex activation function $g_c(z)$, and hidden neuron number K ,

step 1 Randomly choose the complex input weight $\mathbf{w}_k$ and the complex bias

$$b_k, k = 1, \dots, K.$$

step 2 Calculate the complex hidden layer output matrix $\mathbf{H}$.

¹The theoretical analysis such as universal approximation capability of C-ELM is currently under investigation and will appear in a future paper.

5.2. Complex Extreme Learning Machine (C-ELM) Algorithm 99

step 3 Calculate the complex output weight β using formula (6.4).

Many fully complex activation functions proposed by Kim and Adali [42] can be used in our C-ELM. These include circular functions ($\tan(z) = \frac{e^{iz} - e^{-iz}}{i(e^{iz} + e^{-iz})}$, $\sin(z) = \frac{e^{iz} - e^{-iz}}{2i}$), inverse circular functions ($\arctan(z) = \int_0^z \frac{dt}{1+t^2}$, $\arcsin(z) = \int_0^z \frac{dt}{(1-t^2)^{1/2}}$, $\arccos(z) = \int_0^z \frac{dt}{(1-t^2)^{1/2}}$), hyperbolic functions ($\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$, $\sinh(z) = \frac{e^z - e^{-z}}{2}$) and inverse hyperbolic functions ($\operatorname{arctanh}(z) = \int_0^z \frac{dt}{1-t^2}$, $\operatorname{arsinh}(z) = \int_0^z \frac{dt}{(1+t^2)^{1/2}}$), where $z \in C$.

Definition 5.2.1. (pp. 163-169 of [120]) *A matrix $\mathbf{G}$ is the Moore-Penrose generalized inverse of (real or complex) matrix $\mathbf{A}$, if $\mathbf{AGA} = \mathbf{A}$, $\mathbf{GAG} = \mathbf{G}$, $(\mathbf{AG})^* = \mathbf{AG}$, $(\mathbf{GA})^* = \mathbf{GA}$.*

There are several methods to calculate the Moore-Penrose generalized inverse of (real or complex) matrix. These methods may include but are not limited to orthogonal projection, orthogonalization method, iterative method, and Singular Value Decomposition (SVD) [120, 121]. The orthogonalization method and iterative method have their limitations since searching and iteration are used which we wish to avoid in ELM. The orthogonal project method can be used when $\mathbf{H}^* \mathbf{H}$ is nonsingular and $\mathbf{H}^\dagger = (\mathbf{H}^* \mathbf{H})^{-1} \mathbf{H}^*$. However, $\mathbf{H}^* \mathbf{H}$ may not always be nonsingular or may tend to be singular in some applications and thus orthogonal projection method may not perform well in all applications. The Singular Value Decomposition (SVD) can be generally used to calculate the Moore-Penrose generalized inverse of $\mathbf{H}$ in all cases.

5.3 Performance Evaluation

5.3.1 Second order complex nonlinear channel model

The second order complex nonlinear channel model used in Example 2 (Section 3.3.2) is adopted here to evaluate the performance of the C-ELM equalizer with 4-QAM signal. In order to read conveniently, the channel model is still given below:

$$\begin{aligned} y_n &= o_n + 0.2o_n^2 + 0.1o_n^3 + v_n, \quad v_n \sim N(0, 0.01) \\ o_n &= (1 - j0.3434)s_n + (0.5 + j0.2912)s_{n-1} \end{aligned} \quad (5.6)$$

The equalizer input dimension is chosen as 2. As usually done in equalization problems, a decision delay τ is introduced in the equalizer so that at time n the equalizer estimates the input symbol $s_{n-\tau}$ rather than s_n and we set $\tau = 0$. 4-QAM symbol sequence s_n is passed through the channel and the real and imaginary parts of the symbol are valued from the set $\{\pm 0.7\}$. The fully complex activation function of both C-ELM and CBP is chosen as $\operatorname{arcsinh}(z) = \int_0^z \frac{dt}{(1+t^2)^{1/2}}$, where $z = \mathbf{w} \cdot \mathbf{x} + b$. In fact, during our studies we find that CBP with the hyperbolic activation function $\tanh(z)$ does not converge well and produce oscillation in the error but CBP with the activation function $\operatorname{arcsinh}(z)$ converges, however C-ELM works well with both these complex activation functions and many others. The reason may be that CBP gets stuck in local minima easily while ELM tends to reach global minimum

directly. Both the input weight vectors $\mathbf{w}_k$ and biases b_k of the C-ELM² are randomly chosen from a complex area centered at the origin with the radius set as 0.1.

Figure 5.1: Eye diagram of CBP equalizer output (Example 2)

The CBP and C-ELM are trained with 1000 data symbols at 20dB SNR. It is found that the CRBF equalizer trained with such small number of training data cannot classify the testing symbols clearly and thus a higher number (10^4) of training data are used to train CRBF equalizer. The hidden neuron numbers of C-ELM and CBP are set to 10. The CMRAN equalizer obtains 32 hidden neurons at the end of the training process after self growing and pruning neurons during training. CRBF equalizer uses 30 hidden neurons for training and testing, same as in [40].

²Open source codes of the ELM algorithm with different testing cases can be downloaded from: <http://www.ntu.edu.sg/eee/icis/cv/egbhuang.htm>

Figure 5.2: Eye diagram of C-ELM equalizer output (Example 2)

Figure 5.3: Error probability for CRBF, CMRAN, CGAP-RBF, CBP and C-ELM (Example 2)

The eye diagrams for CRBF and CMRAN equalizers output are shown in Fig. 3.9 and Fig. 3.10, respectively. Fig. 5.1 and Fig. 5.2 are the eye diagrams for the output of CBP and C-ELM equalizers. Though all equalizers

5.3. Performance Evaluation103

can separate the output space into four zones, CMRAN equalizer output has some disorder data that cannot be classified correctly.

1000 000 test samples are used for SER calculating at various SNRs. The SER curves for CRBF, CGAP-RBF, CMRAN, CBP and C-ELM equalizers are displayed in Fig. 5.3. CGAP-RBF equalizer can achieve the best SER performance with 30 hidden neurons. With only 10 hidden neurons, C-ELM equalizer can obtain better SER performance than CBP, CRBF and CMRAN equalizers. The training time comparison for five equalizers is given in Table 5.1, which shows that the C-ELM equalizer can achieve a very faster training speed than all other equalizers.

Algorithms	# Neurons	# Training data	Training time (s)	Speedup
C-ELM	10	1000	0.031	1
CBP	10	1000	123.17	3973.23
CGAP-RBF	30	1000	10.06	324.52
CMRAN	32	1000	27.98	902.58
CRBF	30	10 ⁴	47.131	1520.35

Table 5.1: Time comparisons of the five equalizers (a) C-ELM, (b) CBP, (c) CGAP-RBF, (d) CMRAN, (e) CRBF (Example 2)

5.3.2 Third order complex nonlinear channel model

In this section, the well-known complex nonminimum-phase channel model used in Example 3 (Section 3.3.3) is used to evaluate the C-ELM equalizer performance. This equalization model is of order 3 with nonlinear distortion for 4-QAM signaling. The channel output y_n (which is also the input of the equalizer) is given by

$$y_n = o_n + 0.1o_n^2 + 0.05o_n^3 + v_n, \quad v_n \sim \mathcal{N}(0, 0.01) \quad (5.7)$$

$$o_n = (0.34 - i0.27)s_n + (0.87 + i0.43)s_{n-1} + (0.34 - i0.21)s_{n-2}$$

where $\mathcal{N}(0, 0.01)$ means the white Gaussian noise (of the nonminimum-phase channel) with mean 0 and variance 0.01.

Figure 5.4: Channel output distribution (Example 3)

The equalizer input dimension is chosen as 3 and decision delay set $\tau = 1$.

Both the input weight vectors $\mathbf{w}_k$ and biases b_k of the C-ELM are randomly

Figure 5.5: Eye diagram of CRBF equalizer output (Example 3)

chosen from a complex area centered at the origin with the radius set as 0.1.

Figure 5.6: Eye diagram of CMRAN equalizer output (Example 3)

CMRAN, CBP and C-ELM are trained with 1000 data symbols at 16dB SNR and CRBF equalizer with 10000 training samples. The hidden neuron

Figure 5.7: Eye diagram of CBP equalizer output (Example 3)

numbers for CRBF, CBP and C-ELM equalizers are set to 30, 10 and 10, respectively. The CMRAN equalizer built up 22 hidden neurons at the end of the training process.

Figure 5.8: Eye diagram of C-ELM equalizer output (Example 3)

Figure 5.9: Error probability for C-ELM, CMRAN, CRBF and CBP (Example 3)

Fig. 5.4 shows the distribution of the input data of the different equalizers. Fig. 3.15 - Fig. 5.8 are the eye diagram of the outputs of the CRBF, CMRAN, CBP and C-ELM, respectively. As observed from Fig. 5.7 and Fig. 5.8, both C-ELM and CBP can separate the outputs into four regions clearly. Average of 10^6 testing samples at various SNRs were used for computing the symbol error rate (SER) and the comparison of SER for five equalizers is shown in Fig. 5.9. As observed from Fig. 5.9, C-ELM is superior to all other equalizers in terms of SER. Table 5.2 shows the training time comparison for the five equalizers. It can be seen that the C-ELM equalizer can complete training much faster than all other equalizers.

Algorithms	# Neurons	# Training data	Training time (s)	Speedup
C-ELM	10	1000	0.032	1
CBP	10	1000	1.266	39.56
CGAP-RBF	22	1000	17.11	534.69
CMRAN	22	1000	25.481	796.28
CRBF	30	10 ⁴	46.331	1447.84

Table 5.2: Time comparisons of the four equalizers (a) C-ELM, (b) CBP, (c)CGAP-RBF, (d) CMRAN, (e) CRBF (Example 3)

5.4 Discussion

In the above section, C-ELM equalizer is compared with four other equalizers in minimum and nonminimum phase channel models. For second order channel model, CGAP-RBF equalizer can easily partition the input space and achieve the best SER performance by using the extended Kalman filter method which can obtain global optimal solution. While, for third order channel model and the equalizer order is three, CGAP-RBF equalizer’s performance is poorer than that of C-ELM equalizer due to the complicated input space partition. For C-ELM equalizer, the output layer weights are analytically calculated using least square method. Therefore, C-ELM equalizer can achieve good performance than others in terms of SER and learning speed. From the comparison in Table 5.1 and 5.2, it is easily seen that the learning speed of C-ELM is very fast than that of CBP, CRBF, CGAP-RBF

and CMRAN. This is because:

- The input weights and biases of C-ELM are randomly chosen and the output weights are simply analytically calculated, which is much different with the traditional gradient-based learning algorithms. For gradient-based learning algorithms (CBP and CRBF), the network parameters are tuned along the gradient descent direction at each training sample, which will cost much memory and time.
- The traditional gradient-based learning algorithms frequently face several problems like local minimum and improper learning. However, C-ELM can reach the least square solutions straightforward without such issues.

Another issue I want to discuss is how to set proper number of hidden neuron that all fixed structure networks will face. By now, the reasonable solutions are still the designer's experience or trial and error method. The Table 5.3 is listed some results to indicate the effect of the variance of hidden neuron to the network performance for Example 3. The results are gotten after 20 independent runs with 1000 training samples on one P4/2.8GHz personal computer. It can be seen that the training error decreases with the increasing of hidden neurons and the learning speed is really fast even for the network with 100 hidden neurons. However, because of over fitting reason, the symbol error rate (SER) reduces at the beginning and rises after 30

hidden neuron (see Fig. 5.10). When the neuron number of neuron network is beyond the actually required neuron number, the general performance of network will reduce with the increasing of hidden neuron.

#Neurons	Training RMSE		Testing RMSE		Training time		log10(SER)	
	Mean	Dev	Mean	Dev	Mean	Dev	Mean	Dev
10	0.384	0.005	0.388	0.006	0.027	0.045	−2.57	0.006
20	0.367	0.003	0.372	0.004	0.041	0.008	−2.88	0.022
30	0.361	0.005	0.367	0.005	0.065	0.006	−3.00	0.049
40	0.351	0.006	0.363	0.006	0.108	0.005	−2.81	0.059
50	0.334	0.008	0.345	0.008	0.155	0.005	−2.51	0.007
60	0.312	0.006	0.329	0.007	0.205	0.01	−2.51	0.011
70	0.297	0.005	0.316	0.005	0.271	0.017	−2.24	0.059
80	0.284	0.006	0.308	0.007	0.349	0.011	−2.23	0.007
90	0.278	0.005	0.303	0.009	0.432	0.008	−2.41	0.011
100	0.267	0.005	0.293	0.007	0.518	0.011	−2.20	0.007

Table 5.3: Performance comparison of C-ELM equalizer with different hidden neuron (Example 3).

5.5 Conclusions

In this chapter, we propose a fully complex learning algorithm for single-hidden-layer feedforward neural networks (SLFNs) which is referred to as

Figure 5.10: Error probability for different hidden neuron (Example 3)

fully complex extreme learning machine (C-ELM) and its performance has been tested in communication channel equalizers. Similar to ELM [44, 45], the input weights (linking the input layer to the hidden layer) and hidden layer biases of C-ELM are randomly generated and then the output weights (linking the hidden layer to the output layer) are simply analytically calculated instead of iteratively tuned. As observed from the simulation results, the proposed C-ELM can complete the learning phase in an extremely fast speed and obtain much lower symbol error rate (SER). Consistent to the conclusion of Kim and Adali [42] compared to split-complex activation (basis) functions based on neural models (CMRAN and CRBF) the fully complex models (C-ELM and CBP) provide parsimonious structures for applications in the complex domain. It should be noted that as analyzed by Kim and Adali [42] the CBP learning algorithm is sensitive to the size of the learning

rate and the radius of initial random weights and as done in our simulations the learning rate and the radius of initial random weights need to be carefully tuned. Different from other equalizers, C-ELM has avoided well the difficulties in manually tuning control parameters (learning rate, initial weights/biases, learning epochs) and prevented local minima by reaching the good solutions analytically. C-ELM can be implemented and used easily. In fact, faster learning speed, faster response and ease of implementation are the key to the success of the communication channel equalizers. In principle, as tested in our various simulations, many fully complex activation functions introduced by Kim and Adali [42] can be used in the proposed C-ELM and its universal approximation capability will be provided in details in the near future.

Chapter 6

Function Approximation Using Complex Extreme Learning Machine

In the previous chapter, we extend the efficient learning algorithm named extreme learning machine from real domain to the complex domain which can deal with the complex-valued data set. The input, output and hidden neuron activation function of C-ELM network are all complex-valued. The performance evaluation for C-ELM has been done in the communication channel equalization, which can be considered as a kind of classification problems. As we know, function approximation is another very popular issue compared with classification problem. In this chapter, C-ELM algorithm with differ-

ent complex-valued activation functions has been further investigated and applied it for solving the function approximation problem. The comparison with other neural network structures indicates that C-ELM outperforms in terms of approximation accuracy and learning speed.

6.1 Introduction

Split complex sigmoid activation functions consisting of two real-valued activation functions, one for real part and the other for imaginary part, have been traditionally employed in many complex-valued neural networks [34, 36, 62]. The CBP algorithm is used to adapt the network parameters at each training step. Georgiou and Koutsougeras [35] developed a bounded but not entire function and prove that this kind of function is a suitable activation function for CBP algorithm. Kim and Adali [37, 42] have proposed a very important complex neural network model - a fully complex multilayer perceptron (MLP) which uses a class of elementary transcendental function as complex-valued activation function. They have rigorously proved [42] that with very mild condition on the complex activation functions the fully complex MLPs can universally approximate any continuous complex mappings. The corresponding fully complex backpropagation (CBP) learning algorithm with fully complex activation function has also been successfully used in communication applications [37]. But if the interest domain is larger and the initial

random hidden layer weight radius is not small enough, the training process tends to become more sensitive to the size of the learning rate and the radius of initial random weights.

Cha and Kassam [40] have proposed a complex-valued radial basis function (CRBF) network which adopts the stochastic gradient learning algorithm to adjust parameters. Comparing with previously existing equalizers, the CRBF equalizer is superior in terms of symbol error rate (SER). Jianping et al. [41] have developed a complex-valued minimal resource allocation network (CMRAN) equalizer. Applying the growing and pruning criterion, the CMRAN equalizer can realize a more compact structure and can obtain better performance than CRBF and many other equalizers. The Gaussian function used in RBF network is bounded but not entire in complex domain.

The complex extreme learning machine (C-ELM) proposed in the previous chapter has been successfully applied to channel equalization with QAM signals. Compared with traditional methods, C-ELM algorithm has several outstanding characteristics, such as fast learning speed, easy implementation and lower symbol error rate.

The function approximation is to model an unknown function from an input/output pair set that represent the function. Most of the research works have been done in real domain commonly [122, 123, 124, 125, 126]. However, in some application fields such as signal processing, control theory and com-

6.2. C-ELM Algorithm with Different Activation Functions 116

munications, the complex-valued signals are often involved which inspires the function approximation in complex domain [127, 128, 129, 130]. In this chapter, we introduce some complex activation functions that can be applied to C-ELM algorithm and evaluate performance of these different kernel C-ELM algorithms in the complex function approximation.

6.2 C-ELM Algorithm with Different Activation Functions

6.2.1 Brief of complex extreme learning machine

For the sake of readability, this section briefly introduce the complex extreme learning machine (cf. Chapter 5 for details). Given a series of complex-valued training samples $(\mathbf{x}_t, \mathbf{y}_t)$, $t = 1, 2, \dots, N$, the actual outputs of the SLFN network with complex activation function $g_c(z)$ for these N training data is calculated by

$$\begin{cases} \sum_{k=1}^K \beta_k g_c(\mathbf{x}_t, \mathbf{w}_k, b_k) = \mathbf{o}_t, & t = 1, \dots, N, \text{ sigmoid function} \\ \sum_{k=1}^K \beta_k g_c(\mathbf{x}_t, \mu_k, \sigma_k) = \mathbf{o}_t, & t = 1, \dots, N, \text{ RBF function} \end{cases} \quad (6.1)$$

The complex SLFNs with K hidden neurons can approximate the N input samples with zero error means that $\sum_{t=1}^N \|\mathbf{o}_t - \mathbf{y}_t\| = 0$, in other words, the

6.2. C-ELM Algorithm with Different Activation Functions 117

network parameters need to be found such that

$$\sum_{k=1}^K \beta_k g_c(\mathbf{x}_t) = \mathbf{y}_t, \quad t = 1, \dots, N, \quad (6.2)$$

The above N equations can be written compactly as

$$\mathbf{H}\beta = \mathbf{Y} \quad (6.3)$$

where the matrix $\mathbf{H}$ is called hidden layer output matrix. The least-squares solution $\hat{\beta}$ of the linear system $\mathbf{H}\beta = \mathbf{y}$ can be calculated by

$$\hat{\beta} = \mathbf{H}^\dagger \mathbf{Y} \quad (6.4)$$

where matrix $\mathbf{H}^\dagger$ is the Moore-Penrose generalized inverse. The three-step complex ELM algorithm can be summarized as follows:

Algorithm C-ELM: Given a training set $\aleph = \{(\mathbf{x}_t, \mathbf{y}_t) | \mathbf{x}_t \in \mathbf{C}^n, \mathbf{y}_t \in \mathbf{C}^m, t = 1, \dots, N\}$, and hidden neuron number K ,

step 1 Randomly choose the first layer parameters.

step 2 Calculate the hidden layer output matrix $\mathbf{H}$.

step 3 Calculate the complex output weight β using formula (6.4).

6.2.2 Fully complex activation function

We have given the detail discussion about the existed complex activation function in Section 2.1.1. They are split complex activation function, joint

6.2. C-ELM Algorithm with Different Activation Functions 118

nonlinear complex activation function and elementary transcendental functions (ETFs). The formulas for the different complex activation function are listed below:

- split complex activation function

$$f(z) = \frac{1}{1 + \exp(-x)} + i \frac{1}{1 + \exp(-y)} \quad (6.5)$$

where $z = x + iy, z \in C$.

- joint nonlinear complex activation function

$$f(z) = \frac{z}{(c + |z|/r)}, z \in C, c, r \in R \quad (6.6)$$

- elementary transcendental functions

$$f(z) = \operatorname{arcsinh}(z) = \int_0^z \frac{dt}{(1 + t^2)^{1/2}} \quad (6.7)$$

here, for elementary transcendental functions, we just adopt the arcsinh function because it can provide efficient nonlinear approximation capability than others [42].

6.2.3 Radial basis activation function

The radial basis function (Gaussian function) is a common alternative for complex neural network though the output of function is a real value [39, 40,

41]. The output of Gaussian function is given by

$$\begin{aligned} f(\mathbf{x}) &= \exp\left(-\frac{\|\mathbf{x} - \mu\|^2}{\sigma^2}\right) \\ &= \exp\left(-\frac{(\mathbf{x} - \mu)^H (\mathbf{x} - \mu)}{\sigma^2}\right) \end{aligned} \quad (6.8)$$

where $\|\cdot\|$ is the Euclidean distance between the two complex-valued vectors and H is complex conjugate transposition.

6.3 Performance Evaluation

In this section, two approximation problems in complex domain used in [62] are performed to evaluate the approximation capability of complex extreme learning machine and other complex methods such as complex BP (CBP), split BP, complex minimal resource allocated network (CMRAN) and complex RBF (CRBF) network. In the simulations, 1000 training samples and 1000 testing samples are randomly drawn from the interval $[0 + i0, 1 + i]$. Both the input weight vectors $\mathbf{w}_k$ and biases b_k of the C-ELM are randomly chosen from a complex area centered at the origin with the radius set as 0.1. The simulation results are obtained after 20 independent run for each method using a P4/3GHZ personal computer.

6.3.1 Non-analytic function approximation

Example 5: The non-analytic function used in [62] is given by:

$$f(z) = f(x + iy) = e^{iy}(1 - x^2 - y^2)$$

(6.9)

We randomly choose 30 testing data to illustrate the approximation performance for the methods that can obtain satisfying results. Fig. 6.1 to Fig. 6.4 display the constellation of target function output and approximated function output, in which the “o” presents the desired function output and “+” presents the actual output. We can see C-ELM algorithm can achieve very good overlapping performance.

Figure 6.1: Function output using C-ELM with joint activation function (Example 5)

The detail simulation results for Example 5 are reported in Table 6.1. It can be seen that C-ELM algorithm with ETFs activation functions can get

Figure 6.2: Function output using C-ELM with split activation function
(Example 5)

Figure 6.3: Function output using C-ELM with Gaussian activation function
(Example 5)

Figure 6.4: Function output using CMRAN (Example 5)

a least squares solution but not the satisfying one because the self-property limitation of ETFs functions in this case. C-ELM algorithm with joint, split sigmoid and Gaussian function can obtain a very good approximation capabilities and especially Gaussian function can reach 0.0074 training error. Complex BP algorithm with arcsinh and split sigmoid function cannot approximate the target function completely and oscillation is found during the training process. For RBF network method, CGAP-RBF and CMRAN are better than CRBF greatly, CGAP-RBF 0.0314, CMRAN 0.0286 and CRBF 0.3088, respectively. When we compare the training time of different learning methods, C-ELM algorithm outperforms all of them. C-ELM algorithm is hundreds or thousands times faster than other methods no matter what kinds of complex activation functions used.

Figure 6.5: Function output using CGAP-RBF (Example 5)

6.3.2 Analytic function approximation

Example 6: Here, we will use one analytic function to evaluate the approximation capability of different methods. The analytic function is given by:

$$f(z) = f(x + iy) = \sin(x)\cosh(y) + i\cos(x)\sinh(y) \tag{6.10}$$

Algorithms		# Neurons	Training RMSE		Testing RMSE		Training Time (s)	
			Mean	Dev	Mean	Dev	Mean	Dev
C-ELM	arcsinh(z)	10	0.3157	0.003	0.3178	0.002	0.0156	0.005
	Joint function	10	0.0148	0.006	0.0151	0.006	0.0054	0.009
	Split sigmoid	10	0.0111	0.004	0.0112	0.004	0.007	0.008
	Gaussian	10	0.0074	0.001	0.0075	0.002	0.006	0.008
CBP(arcsinh)		10	0.3103	0.078	0.3041	0.055	299.57	10.79
CBP(split)		10	0.2657	0.095	0.2652	0.093	636.76	16.57
CGAP-RBF		10.6	0.0314	0.005	0.0325	0.005	1.73	0.144
CMRAN		9.3	0.0286	0.012	0.030	0.013	1.92	0.087
CRBF		10	0.3088	0.054	0.3074	0.053	0.859	0.031

Table 6.1: Performance comparison for Example 5

6.3. Performance Evaluation

Fig. 6.6 shows the constellation of function output with C-ELM algorithm. C-ELM algorithm with four different activation functions can approximate the target function completely. The CBP algorithm for function output is given in Fig. 6.7. The CBP with arcsinh function can obtain better performance than that with split activation function. The CMRAN algorithm is also efficient for this problem, but the approximation performance is not superior to C-ELM algorithm.

Figure 6.6: Function output using C-ELM algorithm (Example 6)

Table 6.2 shows the simulation results of Example 6, in which all meth-

Figure 6.7: Function output using CBP algorithm (Example 6)

Figure 6.8: Function output using CMRAN (Example 6)

Figure 6.9: Function output using CGAP-RBF (Example 6)

ods except CRBF can obtain the satisfying approximation results. C-ELM algorithm with ETFs activation functions get the best performance for this case, because the approximation function of Example 6 is analytic and the ETFs functions are also analytic unbounded. They are the same class function and can obtain the best approximation performance. C-ELM algorithm with joint, split sigmoid and Gaussian function can also obtain a very satisfying approximation performance in Example 6 and the training errors are 0.0092, 0.0058 and 0.0099, respectively. Complex BP algorithm with arcsinh and split sigmoid function can approximate the target function well for this problem and the accuracy for training is 0.0057 and 0.072. Similarly for RBF network method, CGAP-RBF can achieve almost the same performance as CMRAN and they are both better than CRBF, CGAP-RBF 0.016, CMRAN 0.023 and CRBF 0.2629. The C-ELM algorithm is the fastest method in all

reported methods.

Algorithms		# Neurons	Training RMSE		Testing RMSE		Training Time (s)	
			Mean	Dev	Mean	Dev	Mean	Dev
C-ELM	arcsinh(z)	10	5.81e - 6	6.80e - 6	5.83e - 6	6.85e - 6	0.016	0.003
	Joint function	10	0.0092	0.003	0.0095	0.003	0.0093	0.009
	split Sigmoid	10	0.0058	0.002	0.0059	0.002	0.014	0.005
	Gaussian	10	0.0099	0.001	0.0101	0.002	0.0062	0.008
CBP(arcsinh)		10	0.0057	0.002	0.0060	0.002	278.13	4.53
CBP(split)		10	0.072	0.016	0.073	0.016	635.9	11.13
CGAP-RBF		10.3	0.016	0.005	0.018	0.005	1.72	0.15
CMRAN		10.5	0.023	0.003	0.024	0.003	2.43	0.13
CRBF		10	0.2629	0.049	0.2648	0.048	0.809	0.031

Table 6.2: Performance comparison for Example 6

6.3.3 Multi-input and single output function

Example 7: One multi-input and single output function is given below:

$$f(z1,z2) = z1 - z2 + \cos(z1) + \sin(z2)$$

(6.11)

Table 6.3 is showed the simulation results of different methods for Example 7. C-ELM algorithm can achieve satisfied learning accuracy using any of four activation functions, especially arcsinh function. CBP algorithm can also get good approximation performance in this case, but it needs more learning time than C-ELM algorithm. At the end of training process, CMRAN algorithm builds about 33.1 neurons and its training error is 0.1357. CGAP-RBF uses fewer neurons than CMRAN and gets better approximation accuracy than it. For CRBF algorithm, it normally needs more hidden neuron and training data to obtain promising performance. In order to compare its performance

with that of other methods, 10 hidden neurons and 1000 training data are used, which is same as other methods' setting, so the performance of CRBF algorithm is poor in this example too.

Algorithms		# Neurons	Training RMSE		Testing RMSE		Training Time (s)	
			Mean	Dev	Mean	Dev	Mean	Dev
C-ELM	arcsinh(z)	10	2.60e - 5	9.94e - 6	2.59e - 5	9.90e - 6	0.395	0.009
	Joint function	10	0.0037	0.0005	0.0049	0.0007	0.341	0.012
	Split sigmoid	10	0.0038	0.0005	0.0049	0.0007	0.340	0.011
	Gaussian	10	0.0925	0.0076	0.0934	0.0080	0.005	0.009
CBP(arcsinh)		10	0.0549	0.0263	0.0587	0.0266	289.42	1.57
CBP(split)		10	0.0953	0.0112	0.0953	0.0083	413.66	16.14
CGAP-RBF		17.6	0.0856	0.0050	0.0929	0.0069	5.39	1.36
CMRAN		33.1	0.1357	0.0498	0.141	0.0511	99.43	11.99
CRBF		10	0.2789	0.0387	0.2811	0.0403	0.6094	0.026

Table 6.3: Performance comparison for Example 7

6.3.4 Discussion

In above section, three complex function approximation problems are used to evaluate the performances of C-ELM and other methods. The simulation results show that C-ELM with bounded activation functions (joint function, split sigmoid function and Gaussian function) can achieve satisfied learning accuracy for all simulations. However, C-ELM with $arcsinh(z)$ function (analytic but not bounded function) will faces a problem when it approximates the multi-to-one function (Example3 and Example 5). The reason to explain this issue is still needed to further investigate in the future. No matter what kind of activation functions are chosen, C-ELM can obtain very fast learning

speed, which is because the first layer parameters C-ELM are randomly generated and then simply calculate the output layer weights. All parameters of C-ELM need not to be tuned during the training process which leads to a fast learning speed.

For complex BP algorithm, the oscillation is found during the training process, which causes that CBP algorithm cannot approximate the case of Example 5. We can also found that CBP is the most time consuming algorithm in all the methods listed in our simulation, because CBP (gradient-based learning algorithm) needs much learning epoch to reach a promising learning accuracy. For sequential learning algorithms (CGAP-RBF, CM-RAN and CRBF), CGAP-RBF and CM-RAN achieve good approximation performance in all simulation cases because its growing and pruning criterion can obtain a compact network structure and complex extended Kalman filter method can avoid the local minimum problem. In order to compare the performance with other methods, CRBF used the same hidden neuron number (10 neurons) and training data (1000 data) as that of others. It can be seen that CRBF totally fails to approximate functions in all examples. CRBF network uses gradient-based learning algorithm to adjust the network parameters, which needs more neurons and training data to obtain satisfied approximation performance.

6.4 Conclusions

In this chapter, we discuss the complex activation functions that can be used in C-ELM algorithm and apply C-ELM algorithm for function approximation and equalization problems to evaluate its performance. Similar to ELM [44, 45], the input weights (linking the input layer to the hidden layer) and hidden layer biases of C-ELM are randomly generated and then the output weights (linking the hidden layer to the output layer) are simply analytically calculated instead of iteratively tuned. As observed from the simulation results, C-ELM with $f(z) = z/(1 + |z|)$, split sigmoid and Gaussian activation function can obtain much lower approximation accuracy in function approximation and much lower symbol error rate (SER) in equalization. While C-ELM with ETFs functions sometimes cannot reach a very good general performance in some case (seen Example 5) because of function self-property limitation. No matter what kinds of activation functions, C-ELM completes the learning phase in an extremely fast speed than other methods. It should be noted that as analyzed by Kim and Adali [42] the CBP learning algorithm is sensitive to the size of the learning rate and the radius of initial random weights and as done in our simulations the learning rate and the radius of initial random weights need to be carefully tuned. Different from other equalizers, C-ELM has avoided well the difficulties in manually tuning control parameters (learning rate, initial weights/biases, learning epochs) and prevented local minima by reaching the good solutions analytically.

Chapter 7

Conclusion and Recommendations

7.1 Conclusions

Neural network is a very useful method in many applications, such as control, function approximation, classification, signal processing and system identification. A variety of neural network architectures with different learning algorithms are developed to accomplish different tasks. In this thesis, we focus on investigating the learning algorithm for two neural networks models, radial basis function networks and single layer feedforward networks.

In the first part of the thesis, the extension form of GAP-RBF algorithm in complex domain is proposed in order to process the complex-valued signal

data. One simply formula is derived to evaluate the contribution of each hidden neuron for the overall network output. By linking the significance of hidden neuron to learning accuracy, the growing and pruning strategy are presented to construct a compact network. The complex EKF algorithm is chosen to adjust the network parameters of the nearest hidden neuron. The complex linear and nonlinear channels are used to test the performance of complex-valued GAP-RBF (CGAP-RBF) equalizer in the channel equalization with QAM signal. CGAP-RBF equalizer with less hidden neuron and training samples is able to obtain better SER performance than CMRAN and CRBF equalizers.

In the second part of the thesis, the universal approximation capability of incremental feedforward neural networks in complex domain is theoretically proved. Two complex function approximation problems are chosen to further verify the proof validity. The simulation results show that the proof is correct. After that, the fully complex extreme learning machine (C-ELM) is proposed for communication channel equalization with QAM signal. Different from traditional learning algorithm, C-ELM does not adjust the network parameters during the training process, which leads to a very fast learning speed. C-ELM equalizer can obtain low symbol error rate than CMRAN, CRBF and CBP equalizers. The capability of C-ELM in complex-valued function approximation problems is also investigated in this thesis. C-ELM algorithm with different complex activation function has been used for the

non-analytic and analytic functions. C-ELM with joint, split sigmoid and Gaussian function can achieve very good approximation accuracy. The C-ELM algorithm can obtain the optimal value for the output layer weights directly which can avoid confronting the oscillation problem the tuning-based learning algorithm always met. The easy implementation and fast learning speed make C-ELM algorithm became an efficient method for classification and approximation problems.

7.2 Recommendations for Future Research

Based on the research work presented in this thesis, some suggestions for future research are summarized as following:

- In this thesis, we apply the proposed CGAP-RBF and C-ELM algorithm to communication channel equalization problem. The channel models we used in the simulation are all well known and the channel coefficients do not vary with time. As we know, the channel models are not always fixed in the real application. In particular, the time varying multipath fading that commonly exists in a wireless communication environment leads to severe intersymbol interference. The equalization capability of CGAP-RBF and C-ELM equalizers in fading channel should be investigated.

7.2. Recommendations for Future Research

134

- Though the universal approximation capability of incremental feedforward neural networks in complex domain is theoretically proved and further verified by the simulations, the theoretical proof for other network structures is worthy thoroughly investigating in details in the near future.
- This thesis has evaluated the proposed CGAP-RBF and C-ELM algorithm to channel equalizer and approximation problem. In the future work, it is worthy further studying their performance in other significative applications such as complex time series prediction and complex system identification.

Author's Publications

1. Ming-Bin Li, Guang-Bin Huang, P. Saratchandran and N. Sundararajan, "Performance evaluation of GAP-RBF network in channel equalization," *Neural Processing Letters*, Vol. 22, no. 2, pp. 223-233, 2005.
2. Ming-Bin Li, Guang-Bin Huang, P. Saratchandran and N. Sundararajan, "Fully Complex Extreme Learning Machine," *Neurocomputing*, Vol. 68, pp. 306-314, 2005.
3. Ming-Bin Li, Guang-Bin Huang, P. Saratchandran and N. Sundararajan, "Complex-Valued Growing and Pruning RBF (CGAP-RBF) Neural Networks For Communication Channel Equalization," *IEE Proceedings of Vision, Image and Signal Processing*, Vol.153, no.4, pp.411-418, 2006.
4. Guang-Bin Huang, Ming-Bin Li, Lei Chen and Chee-Kheong Siew, "Incremental Extreme Learning Machine With Fully Complex Hidden Nodes," *accepted by the ISNN2007 special issue of Neurocomputing*.
5. Ming-Bin Li, P. Saratchandran, Guang-Bin Huang and N. Sundararajan, "A Complex-valued Growing and Pruning RBF (CGAP-RBF) Neural Network for Communication Channel Equalization Problems,"

7.2. Recommendations for Future Research

136

- Proceedings of Communication and Computer Networks (CCN)*, MIT, USA, Nov. 8-10, 2004.
6. Ming-Bin Li, Guang-Bin Huang, P. Saratchandran and N. Sundararajan, "Channel Equalization Using Complex Extreme Learning Machine With RBF kernel," *Proceedings, Part III, 114-119, Third International Symposium on Neural networks (ISNN2006)*, Chengdu, China, May 28-June 1, 2006

Bibliography

- [1] C. Lau, *Neural Networks: Theoretical Foundations and Analysis*. Arlington, Virginia, 1992.
- [2] M. H. Hassoun, *Fundamentals of Artificial Neural Networks*. The MIT Press, Cambridge, Massachusetts, 1995.
- [3] A. Hirose, *Complex-valued Neural Networks: Theories and Applications*. Tokyo, Japan, 2003.
- [4] W. S. McCulloch and A. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bulletin of Mathematical Biophysics*, vol. 5, pp. 115–133, 1943.
- [5] P. J. Antsaklis, "Neural networks for control systems," *IEEE Transactions on Neural Networks*, vol. 1, no. 2, pp. 242–244, 1990.
- [6] R. K. Madyastha and B. Aazhang, "An algorithm for training multilayer perceptrons for data classification and function interpolation,"

- IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 41, no. 12, pp. 866–875, 1994.
- [7] B. J. Lee, “Parallel neural networks for speech recognition,” in *International Conference on Neural Networks*, vol. 4, pp. 2093–2097, 9–12 June, 1997.
- [8] S.-W. Lee and H.-H. Song, “A new recurrent neural network architecture for visual pattern recognition,” *IEEE Transactions on Neural Networks*, vol. 8, no. 2, pp. 331–340, 1997.
- [9] C. You and D. Hong, “‘Stop-and-go’ decision-directed blind adaptive equalization using the complex-valued multilayer perceptron,” *IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 4, pp. 21–24, 1997.
- [10] P. K. H. Phua, X. Zhu, and C. H. Koh, “Forecasting stock index increments using neural networks with trust region methods,” in *Proceedings of the International Joint Conference on Neural Networks*, vol. 1, pp. 260–265, 20–24 July, 2003.
- [11] Y. X. H. Leung, N. Xie, and E. Bosse, “A new regression estimator with neural network realization,” *IEEE Transactions on Signal Processing*, vol. 53, no. 2, pp. 672–685, 2005.
- [12] R. P. Brent, “Fast training algorithms for multilayer neural nets,” *IEEE Transaction on Neural Networks*, vol. 2, no. 3, pp. 346–354, 1991.

- [13] C. You and D. Hong, "Nonlinear blind equalization schemes using complex-valued multiplayer feedforward neural networks," *IEEE Transactions on Neural Networks*, vol. 9, no. 6, pp. 1442–1455, 1998.
- [14] S. Guarnieri, F. Piazza, and A. Uncini, "Multilayer feedforward networks with adaptive spline activation function," *IEEE Transactions on Neural Networks*, vol. 10, no. 3, pp. 672–683, 1999.
- [15] H. Roughoopath and S. Roughoopath, "Extended kalman filter learning algorithm for hyper-complex multi-layered neural networks," *International Joint Conference on Neural networks (IJCNN)*, vol. 3, pp. 1824–1828, 1999.
- [16] R. C. Huang and M. S. Chen, "Adaptive equalization using complex-valued multilayered neural network based on the extended kalman filter," *5th International Conference on Signal Processing (WCCC-TCSP)*, vol. 1, pp. 519–524, 2000.
- [17] D. S. Yeung and X. Sun, "Using function approximation to analyze the sensitivity of mlp with antisymmetric squashing activation function," *IEEE Transactions on Neural Networks*, vol. 13, no. 1, pp. 34–44, 2002.
- [18] M. Magdon-Ismail and A. Atiya, "Density estimation and random variate generation using multilayer networks," *IEEE Transactions on Neural Networks*, vol. 13, no. 3, pp. 497–520, 2002.

- [19] S. Chen and B. Mulgrew, "Overcoming co-channel interference using an adaptive radial basis function equaliser," *Signal processing*, vol. 28, pp. 91–107, 1992.
- [20] L. Yingwei, N. Sundararajan, and P. Saratchandran, "Adaptive nonlinear system identification using minimal radial basis function neural networks," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 6, pp. 3521–3524, 1996.
- [21] Q. Gan, N. Sundararajan, and P. Saratchandran, "A complex-valued radial basis function network for equalization of fast time varying channels," *IEEE Transaction on Neural Networks*, vol. 10, no. 4, pp. 958–960, 1999.
- [22] X. Zhou and X. Wang, "Channel estimation for OFDM systems using adaptive radial basis function networks," *IEEE Transactions on Vehicular Technology*, vol. 52, no. 1, pp. 48–59, 2003.
- [23] G.-B. Huang, P. Saratchandran, and N. Sundararajan, "An efficient sequential learning algorithm for growing and pruning RBF (GAP-RBF) networks," *IEEE Transactions on Systems, Man and Cybernetics-Part B: Cybernetics*, vol. 34, no. 6, pp. 2284–2292, 2004.
- [24] J. Platt, "A resource-allocating network for function interpolation," *Neural Computation*, vol. 3, pp. 213–225, 1991.

- [25] V. Kadiramanathan and M. Niranjan, "A function estimation approach to sequential learning with neural networks," *Neural Computation*, vol. 5, pp. 954–975, 1993.
- [26] L. Yingwei, N. Sundararajan, and P. Saratchandran, "A sequential learning scheme for function approximation using minimal radial basis function (RBF) neural networks," *Neural Computation*, vol. 9, pp. 461–478, 1997.
- [27] L. Yingwei, N. Sundararajan, and P. Saratchandran, "Performance evaluation of a sequential minimal radial basis function (RBF) neural network learning algorithm," *IEEE Transactions on Neural Networks*, vol. 9, no. 2, pp. 308–318, 1998.
- [28] G.-B. Huang, Y.-Q. Chen, and H. A. Babri, "Classification ability of single hidden layer feedforward neural networks," *IEEE Transactions on Neural Networks*, vol. 11, no. 3, pp. 799–801, 2000.
- [29] S. Chen, B. Mulgrew, and P. M. Grant, "A clustering technique for digital communications channel equalization using radial basis function networks," *IEEE Transactions on Neural Networks*, vol. 4, no. 4, pp. 570–579, 1993.
- [30] P.-R. Chang and B.-C. Wang, "Adaptive decision feedback equalization for digital satellite channels using multilayer neural networks," *IEEE*

- Journal on Selected Areas in Communications*, vol. 13, no. 2, pp. 316–324, 1995.
- [31] J. Lee, C. Beach, and N. Tepedelenlioglu, “A practical radial basis function equalizer,” *IEEE Journal on Neural Networks*, vol. 10, no. 2, pp. 450–455, 1999.
- [32] S. Abrar, A. Zerguine, and M. Bettayeb, “Recursive least-squares back-propagation algorithm for stop-and-go decision-directed blind equalization,” *IEEE Journal on Neural Networks*, vol. 13, no. 6, pp. 1472–1481, 2002.
- [33] D.-C. Park and T.-K. J. Jeong, “Complex-Bilinear recurrent neural network for equalization of a digital satellite channel,” *IEEE Journal on Neural Networks*, vol. 13, no. 3, pp. 711–725, 2002.
- [34] H. Leung and S. Haykin, “The complex backpropagation algorithm,” *IEEE Transactions on Signal Processing*, vol. 39, no. 9, pp. 2101–2104, 1991.
- [35] G. M. Georgiou and C. Koutsougeras, “Complex domain backpropagation,” *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 39, no. 5, pp. 330–334, 1992.
- [36] T. Nitta, “Extreme learning machine: A new learning scheme of feed-forward neural networks,” in *IEEE International Conference on Neural Networks*, vol. 1, pp. 269–274, 27 June -2 July, 1994.

- [37] T. Kim and T. Adali, "Complex backpropagation neural network using elementary transcendental activation functions," in *Proceedings of IEEE International conf. on Acoustics, Speech, Signal Processing (ICASSP)*, vol. 2, (Salt Lake City, UT), pp. 1281–1284, 2001.
- [38] A. I. Hanna and D. P. Mandic, "A normalised complex backpropagation algorithm," in *Proceedings of IEEE International conf. on Acoustics, Speech, Signal Processing (ICASSP)*, vol. 1, pp. 977–980, 2002.
- [39] S. Chen, S. McLaughlin, and B. Mulgrew, "Complex-valued radial basis function networks, part II: Application to digital communications channel equalization," *Signal Processing*, vol. 36, pp. 165–188, 1994.
- [40] I. Cha and S. A. Kassam, "Channel equalization using adaptive complex radial basis function networks," *IEEE Journal Selected Areas in Communication*, vol. 13, no. 1, pp. 122–131, 1995.
- [41] D. Jianping, N. Sundararajan, and P. Saratchandran, "Communication channel equalization using complex-valued minimal radial basis function neural networks," *IEEE Transactions on Neural Networks*, vol. 13, no. 3, pp. 687–696, 2002.
- [42] T. Kim and T. Adali, "Approximation by fully complex multilayer perceptrons," *Neural Computation*, vol. 15, pp. 1641–1666, July 2003.
- [43] G.-B. Huang, L. Chen, and C.-K. Siew, "Universal approximation using incremental constructive feedforward networks with random hidden

- nodes,” *IEEE Transactions on Neural Networks*, vol. 17, no. 4, pp. 879–892, 2006.
- [44] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, “Extreme learning machine: A new learning scheme of feedforward neural networks,” in *Proceedings of International Joint Conference on Neural Networks (IJCNN2004)*, (Budapest, Hungary), 25-29 July, 2004.
- [45] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, “Extreme learning machine: Theory and applications,” *Submitted to Neurocomputing*, 2005.
- [46] C. Z. W. H. Sweatman, B. Mulgrew, and G. J. Gibson, “Two algorithms for neural-network design and training with application to channel equalization,” *IEEE Transaction on Neural networks*, vol. 9, no. 3, pp. 533–543, 1998.
- [47] Y. Fang and T. W. S. Chow, “Blind equalization of a noisy channel by linear neural network,” *IEEE Transactions on Neural Networks*, vol. 10, no. 4, pp. 918–924, 1999.
- [48] G. Simone and F. C. Morabito, “RBFNN-Based hole identification system in conducting plaes,” *IEEE Transactions on Neural Networks*, vol. 12, no. 6, pp. 1445–1454, 2001.
- [49] F.-J. Chang, J.-M. Liang, and Y.-C. Chen, “Flood forecasting using radial basis function neural networks,” *IEEE Transaction on system, Man, and Cybernetics*, vol. 31, no. 4, pp. 530–535, 2001.

- [50] R. J. Schilling, J. J. Carroll, and A. F. Al-Ajlouni, "Approximation of nonlinear systems with radial basis function neural networks," *IEEE Transaction on Neural networks*, vol. 12, no. 1, pp. 1–15, 2001.
- [51] J. P. Martens and N. Weymaere, "An equalized error backpagation algorithm for the on-line training of multilayer perceptrons," *IEEE Transaction on Neural Networks*, vol. 13, no. 3, pp. 532–541, 2002.
- [52] M. J. Er, S. Wu, J. Lu, and H. L. Toh, "Face recognition with radial basis function (RBF) neural networks," *IEEE Transactions on Neural Networks*, vol. 13, no. 3, pp. 697–710, 2002.
- [53] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of Control, Signals and Systems*, vol. 2, pp. 303–314, 1989.
- [54] K. Hornik, M. Stinchcombe, and H. white, "Multilayer feedforward networks are universal approximation," *Neural Networks*, vol. 2, pp. 359–366, 1989.
- [55] K. L. Ho, Y. Y. Hsu, and C. C. Yang, "Short term load forecasting using a multilayer neural network with an adaptive learning algorithm," *IEEE Transactions on Power Systems*, vol. 7, no. 1, pp. 141–149, 1992.
- [56] T. Parisini and R. Zoppoli, "Neural networks for feedback feedforward nonlinear control systems," *IEEE Transactions on Neural Networks*, vol. 5, no. 3, pp. 436–449, 1994.

- [57] K. D. Rao, "Narrowband direction finding using complex EKF trained multilayered neural networks," *3rd International Conference on Signal processing*, vol. 2, pp. 1377–1380, 1996.
- [58] P. RoyChowdhury, Y. P. Singh, and R. A. Chansarkar, "Dynamic tunneling technique for efficient training of multilayer perceptrons," *IEEE Transactions on Neural Networks*, vol. 10, no. 1, pp. 48–55, 1999.
- [59] W. L. Chan, A. T. P. So, and L. L. Lai, "Initial applications of complex artificial neural networks to load-flow analysis," *IEE Proceedings in Generation, Transmission and Distribution*, vol. 147, no. 6, pp. 361–366, 2000.
- [60] D. L. Lee, "Improving the capacity of complex-valued neural networks with a modified gradient descent learning rule," *IEEE Transaction on Neural Networks*, vol. 12, no. 2, pp. 439–443, 2001.
- [61] D. Srinivasan, J. Xin, and R. L. Cheu, "Evaluation of adaptive neural network models for freeway incident detection," *IEEE Transactions on Intelligent Transportation Systems*, vol. 5, no. 1, pp. 1–11, 2004.
- [62] P. Arena, L. Fortuna, R. Re, and M. G. Xibilia, "Multilayer perceptrons to approximate complex valued functions," *International Journal of Neural Systems*, vol. 6, no. 4, pp. 435–446, 1995.
- [63] T. Kim and T. Adali, "Fully complex backpropagation for constant envelope signal processing," in *Proceedings of IEEE Workshop on Neural*

- Networks for Signal Processing (NNSP)*, vol. 1, (Sydney, Australia), pp. 231–240, 2000.
- [64] S. Chen, S. McLaughlin, and B. Mulgrew, “Complex-valued radial basis function networks, part I: Network architecture and learning algorithms,” *Signal processing*, vol. 35, no. 1, pp. 19–31, 1994.
- [65] N. Sundararajan and P. Saratchandran, “Radial basis function neural networks with sequential learning : MRAN and its applications,” *River Edge: Singapore; World Scientific: NJ*, 1999.
- [66] J. P. Young, T. Hanselmann, A. Zaknich, and Y. Attikiouzel, “Adaptive complex modified probabilistic neural networks in digital channel equalization,” *Intelligent Information Systems Conference, the seventh Australian and New Zealand*, pp. 247–252, 2001.
- [67] G.-B. Huang, P. Saratchandran, and N. Sundararajan, “A generalized growing and pruning RBF (GGAP-RBF) neural network for function approximation,” accepted by *IEEE Transactions on Neural Networks*, 2004.
- [68] J. C. Patra, R. B. Ranendra N. Pal, and G. Panda, “Nonlinear channel equalization for qam constellation using artificial neural networks,” *IEEE Transactions on Systems, Man and Cybernetics*, vol. 29, no. 2, pp. 262–271, 1999.

- [69] D. Jianping, N. Sundararajan, and P. Saratchandran, "complex-valued minimal radial basis function neural network for nonlinear system processing," *International Journal of Neural Systems*, vol. 10, no. 2, pp. 95–106, 2000.
- [70] N. Benvenuto and F. Piazza, "On the complex backpropagation algorithm," *IEEE Transactions on Signal Processing*, vol. 40, no. 4, pp. 967–969, 1992.
- [71] M. R. Smith and H. Yan, "A data extrapolation algorithm using a complex domain neural network," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 44, no. 2, pp. 143–147, 1997.
- [72] S. Chen, P. M. Grant, S. McLaughlin, and B. Mulgrew, "Complex-valued radial basis function networks," in *Third International Conference on Artificial Neural Networks*, pp. 148–152, 25–27 May, 1993.
- [73] B. Igel'nik, M. Tabib-Azar, and S. R. LeClair, "A net with complex weights," *IEEE Transactions on Neural Networks*, vol. 12, no. 2, pp. 236–249, 2001.
- [74] M. Nicolae, C. Botoca, and G. Budur, "Nonlinear complex channel equalization using a radial basis function neural network," in *7th Seminar on Neural Network Applications in Electrical Engineering (NEUREL 2004)*, pp. 73–78, 23–25 Sept, 2004.

- [75] M. Pachter and P. R. Chandler, "Universal linearization concept for extended kalman filters," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 29, no. 3, pp. 946–962, 1993.
- [76] N. U. Ahmed and S. M. Radaideh, "Modified extended kalman filtering," *IEEE Transactions on Automatic Control*, vol. 39, no. 6, pp. 1322–1326, 1994.
- [77] C. Jongsoo, M. Bouchard, and T. H. Yeap, "Decision feedback recurrent neural equalization with fast convergence rate," *IEEE Transactions on Neural Networks*, vol. 16, no. 3, pp. 699–708, 2005.
- [78] K. D. Rao, M. N. S. Swamy, and E. I. Plotkin, "Complex EKF neural network for adaptive equalization," in *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS 2000)*, pp. 349–352, 28-31 May, 2000.
- [79] P. H. G. Coelho, "A complex EKF-RTRL neural network," in *Proceedings of International Joint Conference on Neural Networks (IJCNN 2001)*, pp. 120–125, 15-19 July, 2001.
- [80] C. Beare, "The choice of the desired impulse response in combined linear-viterbi algorithm equalizers," *IEEE Transactions on Communications*, vol. 26, no. 8, pp. 1301–1307, 1978.
- [81] P. E. Ostling, "Performance of MMSE linear equalizer and decision feedback equalizer in single-frequency simulcast environments," in

IEEE 43rd Vehicular Technology Conference, pp. 629–632, 18–20 May, 1993.

- [82] C. Tellambura, Y. J. Guo, and S. K. Barton, “A noniterative approach for computing linear equaliser tap weights,” in *The 8th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, vol. 2, pp. 302–306, 1–4 Sept, 1997.
- [83] F. Riera-Palou, J. M. Noras, and D. G. M. Cruickshank, “Linear equalisers with dynamic and automatic length selection,” *Electronics Letters*, vol. 37, no. 25, pp. 1553–1554, 2001.
- [84] A. Hedayat, A. Nosratinia, and N. Al-Dhahir, “Linear equalizers for flat rayleigh MIMO channels,” in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2005)*, vol. 3, pp. 445–448, 18–23 March, 2005.
- [85] G. Kechriotis, E. Zervas, and E. S. Manolakos, “Using recurrent neural networks for adaptive communication channel equalization,” *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 267–278, 1994.
- [86] F. Mazzenga, “Channel estimation and equalization for M-QAM transmission with hidden pilot sequence,” *IEEE Transaction on broadcasting*, vol. 46, no. 2, pp. 170–176, 2000.

- [87] D. George, R. Bowen, and J. Storey, "An adaptive decision feedback equalizer," *IEEE Transactions on Communications*, vol. 19, no. 3, pp. 281–293, 1971.
- [88] S. Siu, G. J. Gibson, and C. F. N. Cowan, "Decision feedback equalisation using neural network structures and performance comparison with standard architecture," in *IEE Proceedings In Communications, Speech and Vision*, vol. 137, pp. 221–225, Aug, 1990.
- [89] M. Sternad, A. Ahlen, and E. Lindskog, "Robust decision feedback equalizers," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 1993)*, vol. 3, pp. 555–558, 27–30 April, 1993.
- [90] K. Berberidis, T. A. Rontogiannis, and S. Theodoridis, "Efficient block implementation of the decision feedback equalizer," *IEEE Signal Processing Letters*, vol. 5, no. 6, pp. 129–131, 1998.
- [91] J. Tsimbinos and L. B. White, "Decision feedback equalizers for high speed data communications," in *Proceedings of the Fifth International Symposium on Signal Processing and Its Applications*, vol. 2, pp. 765–768, 22–25 Aug, 1999.
- [92] E. G. Lee, J. A. Bastian, M. N. Miyatake, and H. P. Meana, "Decision feedback equalizers for high speed data communications," in *The IVth*

- International Conference on Antenna Theory and Techniques*, vol. 1, pp. 341–344, 9–12 Sept, 2003.
- [93] C. Sheng, S. McLaughlin, B. Mulgrew, and P. M. Grant, “Adaptive bayesian decision feedback equalizer for dispersive mobile radio channel,” *IEEE Transactions on Communications*, vol. 43, no. 5, pp. 1937–1946, 1995.
- [94] C. Sheng, S. McLaughlin, B. Mulgrew, and P. M. Grant, “Bayesian decision feedback equaliser for overcoming co-channel interference,” *IEE Proceedings in Communications*, vol. 143, no. 4, pp. 219–225, 1996.
- [95] C. Tianping and C. Hong, “Approximation capability to functions of several variables, nonlinear functionals, and operators by radial basis function neural networks,” *IEEE Transactions on Neural Networks*, vol. 6, no. 4, pp. 904–910, 1995.
- [96] C. Tianping and C. Hong, “Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems,” *IEEE Transactions on Neural Networks*, vol. 6, no. 4, pp. 911–917, 1995.
- [97] P. Arena, L. Fortuna, R. Re, and M. G. Xibilia, “On the capability of neural networks with complex neurons in complex valued functions approximation,” in *IEEE International Symposium on Circuits and Systems (ISCAS 1993)*, vol. 4, pp. 2168–2171, 3–6 May, 1993.

-
- [98] A. Uncini, L. Vecci, P. Campolucci, and F. Piazza, "Complex-valued neural networks with adaptive spline activation function for digital radio links nonlinear equalization," *IEEE Transactions on Signal Processing*, vol. 47, no. 2, pp. 505–514, 1999.
- [99] H. Jiang and K. S. Kwak, "A new complex recurrent neural network equalizer for data communication," in *Proceedings of the 2002 IEEE International Symposium on Industrial Electronics*, vol. 4, pp. 1271–1275, 8–11 July, 2002.
- [100] S. Chen, B. Mulgrew, and S. McLaughlin, "Adaptive bayesian equalizer with decision feedback," *IEEE Transaction on Signal Processing*, vol. 41, no. 9, pp. 2918–2927, 1993.
- [101] V. Weerackody and S. A. Kassam, "Dual-mode type algorithms for blind equalization," *IEEE Transaction on Communications*, vol. 42, no. 1, pp. 22–28, 1994.
- [102] J. G. Proakis, *Digital Communications*. New York: McGrawHill, 1983.
- [103] D. F. Specht, "Generation of polynomial discriminant functions for pattern recognition," *IEEE Trans. Electron. Comput.*, vol. 16, pp. 308–319, 1967.
- [104] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. New York: Wiley, 1973.

- [105] S. Chen, G. J. Gibson, C. F. N. Cowan, and P. M. Grant, "Adaptive equalization of finite non-linear channels using multilayer perceptrons," *Signal Processing*, vol. 20, pp. 107–119, 1990.
- [106] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, pp. 359–366, 1989.
- [107] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural Networks*, vol. 4, pp. 251–257, 1991.
- [108] J. Park and I. W. Sandberg, "Universal approximation using radial-basis-function networks," *Neural Computation*, vol. 3, pp. 246–257, 1991.
- [109] M. Leshno, V. Y. Lin, A. Pinkus, and S. Schocken, "Multilayer feedforward networks with a nonpolynomial activation function can approximate any function," *Neural Networks*, vol. 6, pp. 861–867, 1993.
- [110] A. R. Barron, "Universal approximation bounds for superpositions of a sigmoid function," *IEEE Transactions on Information Theory*, vol. 39, no. 3, pp. 930–945, 1993.
- [111] G.-B. Huang and C.-K. Siew, "Extreme learning machine: RBF network case," in *Proceedings of the Eighth International Conference on Control, Automation, Robotics and Vision (ICARCV 2004)*, (Kunming, China), 6-9 Dec, 2004.

- [112] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Real-time learning capability of neural networks," *IEEE Transactions on Neural Networks*, vol. 17, no. 4, pp. 863–878, 2006.
- [113] G.-B. Huang and C.-K. Siew, "Extreme learning machine with randomly assigned RBF kernels," *International Journal of Information Technology*, vol. 11, March 2005.
- [114] T. Kim and T. Adali, "Universal approximation of fully complex feed-forward neural networks," in *International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, pp. I-973–I-976, 2002.
- [115] W. L. Voxman and J. Roy H. Goetschel, *Advanced Calculus: An Introduction to Modern Analysis*. Marcel Dekker, Inc., 1981.
- [116] Y. Eidelman, V. Milman, and A. Tsolomitis, *Functional Analysis: An Introduction*. American Mathematical Society, 2004.
- [117] A. N. Kolmogorov and S. V. Fomin, *Elements of the Theory of Functions and Functional Analysis, Volume 2: Measure, The Lebesgue Integral, Hilbert Space*. Graylock Press, Baltimore, MD, 1961.
- [118] S. Tamura and M. Tateishi, "Capabilities of a four-layered feedforward neural network: Four layers versus three," *IEEE Transactions on Neural Networks*, vol. 8, pp. 251–255, March 1997.

-
- [119] G.-B. Huang, "Learning capability and storage capacity of two-hidden-layer feedforward networks," *IEEE Transactions on Neural Networks*, vol. 14, pp. 274–281, March 2003.
 - [120] J. M. Ortega, *Matrix Theory*. Plenum Press, New York and London, 1987.
 - [121] C. R. Rao and S. K. Mitra, *Generalized Inverse of Matrices and its Applications*. New York: John Wiley, 1971.
 - [122] T. Draelos and D. Hush, "A constructive neural network algorithm for function approximation," in *IEEE International Conference on Neural Networks*, vol. 1, pp. 50–55, 3-6 June, 1993.
 - [123] P. K. Simpson and G. Jahns, "Fuzzy min-Max neural networks for function approximation," in *IEEE International Conference on Neural Networks*, vol. 3, pp. 1967–1972, 28 March-1 April, 1993.
 - [124] E. Gelenbe, Z.-H. Mao, and Y.-D. Li, "Function approximation with spiked random networks," *IEEE Transactions on Neural Networks*, vol. 10, no. 1, pp. 3–9, 1999.
 - [125] J. González, I. Rojas, H. Pomares, J. Ortega, and A. Prieto, "A new clustering technique for function approximation," *IEEE Transaction on Neural networks*, vol. 13, no. 1, pp. 132–142, 2002.
 - [126] W.-Y. Wang, T.-T. Lee, C.-L. Liu, and C.-H. Wang, "Function approximation using fuzzy neural networks with robust learning algorithm,"

- IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 27, no. 4, pp. 740–747, 1997.
- [127] N. PNagai and M. Suzuki, “Complex wave digital filters for the analysis of complex signals,” in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 1986)*, vol. 11, pp. 2603–2606, Apr, 1986.
- [128] P. Arena, L. Fortuna, and M. G. Xibilia, “Predicting complex chaotic time series via complex-valued MLPs,” in *IEEE International Symposium on Circuits and Systems (ISCAS 1994)*, vol. 6, pp. 29–32, 30 May–2 June, 1994.
- [129] V. Calhoun and T. Adali, “Complex infomax: Convergence and approximation of infomax with complex nonlinearities,” in *Proceedings of the 2002 12th IEEE Workshop on Neural Networks for Signal Processing*, pp. 307–316, 4–6 Sept, 2002.
- [130] G. S. Lee and D. P. Mandic, “Nonlinear adaptive prediction of complex-valued signals by complex-valued PRNN,” *IEEE Transactions on Signal Processing*, vol. 53, no. 5, pp. 1827–1836, 2005.